# ISP1763A

## Hi-Speed USB OTG controller

Rev. 02 — 24 February 2011 Product data sheet

## 1. General description

The ISP1763A is a single-chip Hi-Speed Universal Serial Bus (USB) On-The-Go (OTG) controller integrated with the advanced ST-Ericsson's slave host controller and the ST-Ericsson's ISP1582 peripheral controller.

The Hi-Speed USB host controller and peripheral controller comply with *Universal Serial Bus Specification Rev. 2.0* and support data transfer speeds of up to 480 Mbit/s. The Enhanced Host Controller Interface (EHCI) core implemented in the host controller is adapted from *Enhanced Host Controller Interface Specification for Universal Serial Bus Rev. 1.0*. The OTG controller is compliant with *On-The-Go Supplement to the USB Specification Rev. 1.3*.

The ISP1763A has two USB ports. Port 1 can be configured to function as a downstream port, an upstream port, or as an OTG port; port 2 is always configured as a downstream port. Port 2 supports Session Request Protocol (SRP) detection from the B-device. The OTG port supports Host Negotiation Protocol (HNP) and SRP as specified in *On-The-Go Supplement to the USB Specification Rev. 1.3*.

The ISP1763A support multiple bus interfaces with 8-bit or 16-bit bus. The ISP1763A can interface to processors with digital I/O voltages of 1.8 V or 3.3 V.

## 2. Features

- Compliant with:
  - *Universal Serial Bus Specification Rev. 2.0*
  - *On-The-Go Supplement to the USB Specification Rev. 1.3*
- Small form-factor for portable applications; available in VFQFPN64 and TFBGA64 Restriction of Hazardous Substances (RoHS) compliant, halogen-free and lead-free packages
- Low power consumption for portable applications
- Host supports data transfer at high-speed (480 Mbit/s), full-speed (12 Mbit/s), and low-speed (1.5 Mbit/s); supports disabling of high-speed mode on each port
- Peripheral supports data transfer at high-speed (480 Mbit/s) and full-speed (12 Mbit/s)
- Integrated Transaction Translator (TT) for Original USB (full-speed and low-speed) support
- Two USB ports:
  - Port 1: OTG, host, or peripheral port
  - Port 2: Host port only (supports SRP detection)
- Supports OTG HNP and SRP
- Supports 8-bit or 16-bit CPU bus interface
- Flexibility to interface with various types of processors:

- NOR Flash interface (multiplexed mode)
- NAND Flash interface (multiplexed mode)
- General multiplex interface
- SRAM interface

- Single configurable interrupt (INT) line for the host controller, peripheral controller, and OTG controller
- Integrated Phase-Locked Loop (PLL) supports external 12 MHz, 19.2 MHz, and 24 MHz crystal, and direct clock source
- Supports remote wake-up from deep sleep mode
- Supports interfacing I/O voltage of 1.8 V or 3.3 V; separate I/O voltage supply pins minimize crosstalk
- Internal voltage regulator supplies 1.2 V to the digital core
- 3.0 V to 3.6 V supply voltage input range for the internal USB transceiver
- Supports hybrid power mode; $V_{CC(3V3)}$ is not present, $V_{CC(I/O)}$ is powered
- Host controller-specific features:
  - EHCI core is adapted from *Enhanced Host Controller Interface Specification for Universal Serial Bus Rev. 1.0*
  - Integrated TT for Original USB device support on both the ports
  - Integrated 24 kB high-speed memory
  - Power switching and overcurrent reporting on per-port basis
- Peripheral controller-specific features:
  - Compliant with *Universal Serial Bus Specification Rev. 2.0*
  - Integrated 4 kB memory to support seven IN endpoints, seven OUT endpoints, and one fixed control IN/OUT endpoint
  - $V_{BUS}$ detection in deep sleep mode
- OTG controller-specific features:
  - Supports OTG HNP and SRP using status and control registers for the software implementation in OTG dual-role devices
  - Integrated $V_{BUS}$ voltage comparators
  - Integrated cable (ID) detector
  - Programmable timers with high resolution (0.01 ms to 80 ms)

# 3. Applications

The ISP1763A can be used to implement a dual-role USB device in any application, USB host or USB peripheral, depending on the cable connection. If the dual-role device is connected to a USB peripheral, it behaves like a USB host. The dual-role device can also be connected to a PC or any other USB host, and behave like a USB peripheral.

## 3.1 Host or peripheral roles

- TV/TV box:
  - Play, upload, or download media files from or to USB memory disk
- DVD player:
  - Play, upload, or download media files from or to USB memory disk
- Mobile phone to or from:

- Mobile phone: exchange contact information
- Digital still camera: e-mail pictures or upload pictures to the web
- MP3 player: upload or download/broadcast music
- Mass storage: upload or download files
- Scanner: scan business cards

- Printer
- Netbook
- Set-top box

# 4. Ordering information

**Table 1. Ordering information**

| Commercial product code | Package description | Packing | Minimum sellable quantity |
|---|---|---|---|
| ISP1763AETTM | TFBGA64; 64 balls; body 4 × 4 × 0.8 mm | 13 inch tape and reel non-dry pack | 4000 pieces |
| ISP1763AHNUM | VFQFPN64; 64 terminals; body 9 × 9 × 1.0 mm | 13 inch tape and reel dry pack | 1000 pieces |

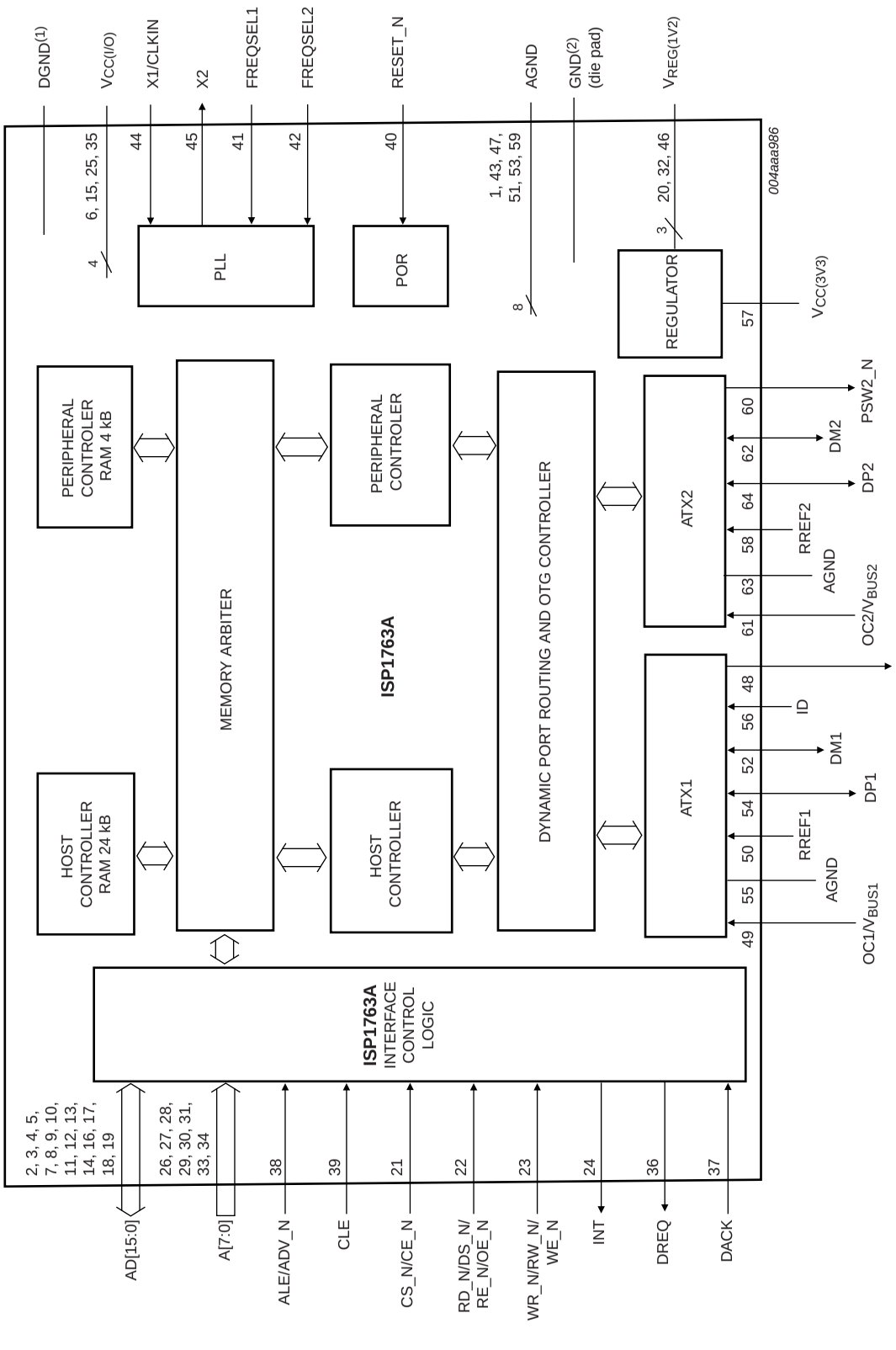
# 5. Marking

**Table 2. Marking codes**

| Type number | Marking code[1] |
|---|---|
| ISP1763AETTM | 1763A |
| ISP1763AHNUM | 1763A |

[1] The package marking is the first line of text on the IC package and can be used for IC identification.

# 6. Block diagram




The figure shows the VFQFPN pinout. For the TFBGA ballout, see Table 3.

(1) Only applicable to the TFBGA package.

(2) Only applicable to the VFQFPN package.

**Fig 1. Block diagram**

# 7. Pinning information

## 7.1 Pinning

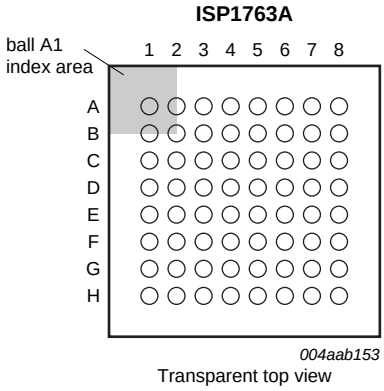



**Fig 2. Pin configuration TFBGA64**

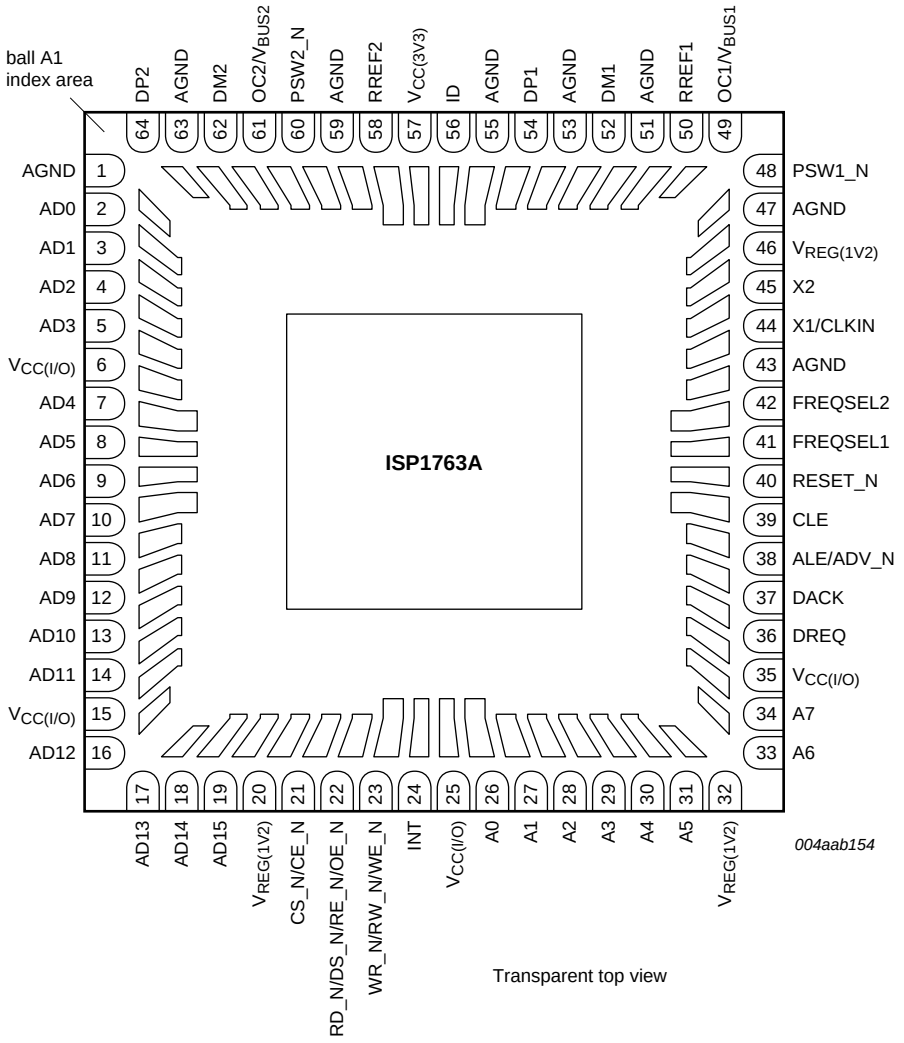



Fig 3. Pin configuration VFQFPN64

## 7.2 Pin description

Table 3. Pin description

| Symbol[1] | Pin | | Type[2] | Description |
|---|---|---|---|---|
| | TFBGA64 | VFQFPN64 | | |
| AGND | B2 | 1 | P | analog ground |
| AD0 | B1 | 2 | I/O | bit 0 of the address and data bus<br>bidirectional pad; push-pull input; three-state output; 3.3 V tolerant |
| AD1 | C2 | 3 | I/O | bit 1 of the address and data bus<br>bidirectional pad; push-pull input; three-state output; 3.3 V tolerant |
| AD2 | C1 | 4 | I/O | bit 2 of the address and data bus<br>bidirectional pad; push-pull input; three-state output; 3.3 V tolerant |

**Table 3. Pin description** *...continued*

| Symbol[1] | Pin | | Type[2] | Description |
|---|---|---|---|---|
| | TFBGA64 | VFQFPN64 | | |
| AD3 | C3 | 5 | I/O | bit 3 of the address and data bus<br>bidirectional pad; push-pull input; three-state output; 3.3 V tolerant |
| $V_{CC(I/O)}$ | D2 | 6 | P | I/O supply voltage; connect a 0.1 μF decoupling capacitor |
| AD4 | D1 | 7 | I/O | bit 4 of the address and data bus<br>bidirectional pad; push-pull input; three-state output; 3.3 V tolerant |
| AD5 | C4 | 8 | I/O | bit 5 of the address and data bus<br>bidirectional pad; push-pull input; three-state output; 3.3 V tolerant |
| AD6 | E1 | 9 | I/O | bit 6 of the address and data bus<br>bidirectional pad; push-pull input; three-state output; 3.3 V tolerant |
| AD7 | E2 | 10 | I/O | bit 7 of the address and data bus<br>bidirectional pad; push-pull input; three-state output; 3.3 V tolerant |
| AD8 | F1 | 11 | I/O | bit 8 of the address and data bus<br>bidirectional pad; push-pull input; three-state output; 3.3 V tolerant |
| AD9 | D3 | 12 | I/O | bit 9 of the address and data bus<br>bidirectional pad; push-pull input; three-state output; 3.3 V tolerant |
| AD10 | D4 | 13 | I/O | bit 10 of the address and data bus<br>bidirectional pad; push-pull input; three-state output; 3.3 V tolerant |
| AD11 | G1 | 14 | I/O | bit 11 of the address and data bus<br>bidirectional pad; push-pull input; three-state output; 3.3 V tolerant |
| $V_{CC(I/O)}$ | F2 | 15 | P | I/O supply voltage; connect a 0.1 μF decoupling capacitor |
| AD12 | E3 | 16 | I/O | bit 12 of the address and data bus<br>bidirectional pad; push-pull input; three-state output; 3.3 V tolerant |
| AD13 | H1 | 17 | I/O | bit 13 of the address and data bus<br>bidirectional pad; push-pull input; three-state output; 3.3 V tolerant |
| AD14 | G2 | 18 | I/O | bit 14 of the address and data bus<br>bidirectional pad; push-pull input; three-state output; 3.3 V tolerant |
| AD15 | H2 | 19 | I/O | bit 15 of the address and data bus<br>bidirectional pad; push-pull input; three-state output; 3.3 V tolerant |
| $V_{REG(1V2)}$ | H3 | 20 | P | core power 1.2 V; connect a 4.7 μF decoupling capacitor either on this pin or on pin 46 (ball C7) |

**Table 3. Pin description** *…continued*

| Symbol[1] | Pin | | Type[2] | Description |
|---|---|---|---|---|
| | TFBGA64 | VFQFPN64 | | |
| CS_N/CE_N | G3 | 21 | I | chip select<br>input; 3.3 V tolerant |
| RD_N/DS_N/ RE_N/OE_N | F3 | 22 | I | read enable, write, or read latch; when not in use, connect to $V_{CC(I/O)}$<br>input; 3.3 V tolerant |
| WR_N/RW_N/ WE_N | H4 | 23 | I | write enable; when not in use, connect to $V_{CC(I/O)}$<br>input; 3.3 V tolerant |
| INT | G4 | 24 | O | interrupt output<br>**Remark:** When the ISP1763A is in power-down mode, the INT pin is a high-impedance I/O. External pull-down resistors are recommended to minimize the I/O leakage.<br>push-pull output; 3.3 V tolerant |
| $V_{CC(I/O)}$ | F4 | 25 | P | I/O supply voltage; connect a 0.1 μF decoupling capacitor |
| A0 | H5 | 26 | I | bit 0 of the address bus; when not in use, connect to GND<br>input; 3.3 V tolerant |
| A1 | G5 | 27 | I | bit 1 of the address bus; when not in use, connect to GND<br>input; 3.3 V tolerant |
| A2 | H6 | 28 | I | bit 2 of the address bus; when not in use, connect to GND<br>input; 3.3 V tolerant |
| A3 | E4 | 29 | I | bit 3 of the address bus; when not in use, connect to GND<br>input; 3.3 V tolerant |
| A4 | H7 | 30 | I | bit 4 of the address bus; when not in use, connect to GND<br>input; 3.3 V tolerant |
| A5 | G6 | 31 | I | bit 5 of the address bus; when not in use, connect to GND<br>input; 3.3 V tolerant |
| $V_{REG(1V2)}$ | H8 | 32 | P | core power 1.2 V input; for normal operation, this pin must be connected to pin 20 or pin 46 for the VFQFPN64 package and ball H3 or ball C7 for the TFBGA64 package |
| A6 | F5 | 33 | I | bit 6 of the address bus; when not in use, connect to GND<br>input; 3.3 V tolerant |
| A7 | G7 | 34 | I | bit 7 of the address bus; when not in use, connect to GND<br>input; 3.3 V tolerant |
| $V_{CC(I/O)}$ | F6 | 35 | P | I/O supply voltage; connect a 0.1 μF decoupling capacitor |
| DREQ | G8 | 36 | O | DMA request; when not in use, pull-down to GND through a 10 kΩ resistor<br>**Remark:** When the ISP1763A is in power-down mode, the DREQ pin is a high-impedance I/O. External pull-down resistors are recommended to minimize the I/O leakage.<br>push-pull output; 3.3 V tolerant |
| DACK | F7 | 37 | I | DMA acknowledge; when not in use, connect to GND through a 10 kΩ resistor<br>input; 3.3 V tolerant |

**Table 3. Pin description** *...continued*

| Symbol[1] | Pin | | Type[2] | Description |
|---|---|---|---|---|
| | TFBGA64 | VFQFPN64 | | |
| ALE/ADV_N | F8 | 38 | I | address latch enable<br>input; 3.3 V tolerant |
| CLE | E6 | 39 | I | command latch enable<br>input; 3.3 V tolerant |
| RESET_N | E7 | 40 | I | internal regulator power-down control; when not in use, connect to $V_{CC(I/O)}$<br>input; 3.3 V tolerant |
| FREQSEL1 | E8 | 41 | I | input clock frequency selection pin 1<br>input; 3.3 V tolerant |
| FREQSEL2 | D8 | 42 | I | input clock frequency selection pin 2<br>input; 3.3 V tolerant |
| AGND | D7 | 43 | P | analog ground |
| X1/CLKIN | C8 | 44 | AI | crystal oscillator or clock input; 1.2 V peak input allowed |
| X2 | B8 | 45 | AO | crystal oscillator output; leave open if an external clock is applied on pin X1/CLKIN |
| $V_{REG(1V2)}$ | C7 | 46 | P | core power 1.2 V; connect a 4.7 μF decoupling capacitor either on this pin or on pin 20 (ball H3) |
| AGND | - | 47 | P | analog ground |
| DGND | E5 | - | P | digital ground |
| PSW1_N | D6 | 48 | OD | port 1 power switch; when not in use, connect to $V_{CC(3V3)}$ through a 10 kΩ resistor<br>open-drain output; 5 V tolerant |
| OC1/$V_{BUS1}$ | A8 | 49 | AI/O | • overcurrent input (OC1) for the host functionality; an external power switch is used<br>• $V_{BUS1}$ for the OTG and peripheral functionality; connected to $V_{BUS}$ detectors, and $V_{BUS}$ SRP charge and discharge circuit.<br>When not in use, connect to $V_{CC(3V3)}$ through a 10 kΩ resistor.<br>5 V tolerant |
| RREF1 | A7 | 50 | AI | port 1 reference resistor connection; see Section 8.12.4 |
| AGND | B7 | 51 | P | analog ground |
| DM1 | A6 | 52 | AI/O | port 1 DM; connect to the D− pin of the USB connector |
| AGND | B6 | 53 | P | analog ground |
| DP1 | A5 | 54 | AI/O | port 1 DP; connect to the D+ pin of the USB connector |
| AGND | C6 | 55 | P | port 1 analog GND |
| ID | B5 | 56 | AI | port 1 ID pin (an internal pull-up resistor is present on this pin) |
| $V_{CC(3V3)}$ | A4 | 57 | P | supply voltage |
| RREF2 | A3 | 58 | AI | port 2 reference resistor connection; see Section 8.12.4 |
| AGND | C5 | 59 | P | analog ground |
| PSW2_N | B4 | 60 | OD | port 2 power switch; when not in use, connect to $V_{CC(3V3)}$ through a 10 kΩ resistor<br>open-drain output; 5 V tolerant |

**Table 3. Pin description** *…continued*

| Symbol[1] | Pin | | Type[2] | Description |
|---|---|---|---|---|
| | TFBGA64 | VFQFPN64 | | |
| OC2/$V_{BUS2}$ | D5 | 61 | AI | port 2 overcurrent input or $V_{BUS}$ detection; when not in use, connect to $V_{CC(3V3)}$ through a 10 kΩ resistor<br>5 V tolerant |
| DM2 | A2 | 62 | AI/O | port 2 DM; connect to the D− pin of the USB connector |
| AGND | B3 | 63 | P | port 2 analog ground |
| DP2 | A1 | 64 | AI/O | port 2 DP; connect to the D+ pin of the USB connector |
| GND | - | die pad | P | ground supply; down bonded to the exposed die pad (heat sink); to be connected to the PCB ground |

[1] Symbol names ending with underscore N (for example, NAME_N) represent active-LOW signals.

[2] I = input only; O = output only; I/O = digital input/output; OD = open-drain output; AI = analog input; AO = analog output; AI/O = analog input/output; P = power or ground.

# 8. Functional description

## 8.1 CPU bus interface

The ISP1763A has a fast advance general-purpose interface to communicate with most types of microcontrollers and microprocessors. This microcontroller interface is configured using pins ALE/ADV_N and CLE to accommodate most types of interfaces. The bus interface supports 8-bit and 16-bit, which can be configured using bit DATA_BUS_WIDTH. Four bus interface types are selected using inputs ALE/ADV_N and CLE during power-up, the RD_N/DS_N/RE_N/OE_N and CS_N/CE_N pins, or the RESET_N pin. Table 4 provides details of bus configurations for each mode.

**Table 4. Bus configuration modes**

| Bus mode | ALE/ADV_N | CLE | DATA_BUS_WIDTH | Signal description |
|---|---|---|---|---|
| SRAM 8-bit | HIGH | HIGH | 1 | • A[7:0]: 8-bit address bus<br>• AD[7:0]: 8-bit data bus<br>• Write (WR_N), read (RD_N), chip select (CS_N): control signals for normal SRAM mode<br>• Write (WR_N/RW_N), data strobe (DS_N), chip select (CS_N): control signals for proprietary SRAM mode (see Figure 14)<br>• DACK: DMA acknowledge input<br>• DREQ: DMA request output |
| SRAM 16-bit | HIGH | HIGH | 0 | • A[7:0]: 8-bit address bus<br>• AD[15:0]: 16-bit data bus<br>• Write (WR_N), read (RD_N), chip select (CS_N): control signals<br>• Write (WR_N/RW_N), data strobe (DS_N), chip select (CS_N): control signals for proprietary SRAM mode (see Figure 14)<br>• DACK: DMA acknowledge input<br>• DREQ: DMA request output |
| NAND 8-bit | LOW | LOW | 1 | • AD[7:0]: 8-bit data bus<br>• ALE, CLE, write enable, read enable (RE_N), chip enable: control signals |
| NAND 16-bit | LOW | LOW | 0 | • AD[15:0]: 16-bit data bus<br>• ALE, CLE, write enable, read enable, chip enable: control signals |
| NOR 8-bit | HIGH | LOW | 1 | • AD[7:0]: 8-bit data bus<br>• ADV_N, write enable, output enable, chip select: control signals |

**Table 4. Bus configuration modes** *...continued*

| Bus mode | ALE/ADV_N | CLE | DATA_BUS_WIDTH | Signal description |
|---|---|---|---|---|
| NOR 16-bit | HIGH | LOW | 0 | • AD[15:0]: 16-bit data bus<br>• ADV_N, write enable, output enable, chip select: control signals |
| General multiplex 8-bit | LOW | HIGH | 1 | • AD[7:0]: 8-bit data bus<br>• ALE, write (WR_N), read (RD_N), chip select: control signals |
| General multiplex 16-bit | LOW | HIGH | 0 | • AD[15:0]: 16-bit data bus<br>• ALE, write (WR_N), read (RD_N), chip select: control signals |

**Table 5. Pinning information of the bus interface**

| SRAM mode | NAND mode | NOR mode | General multiplex mode | Type | Description |
|---|---|---|---|---|---|
| AD[15:0] | AD[15:0] | AD[15:0] | AD[15:0] | I/O | data or address bus |
| A[7:0] | - | - | - | I | address bus |
| - | ALE | ADV_N | ALE | I | address or command valid |
| - | CLE | - | - | I | address or command valid |
| CS_N | CE_N | CS_N | CS_N | I | chip select |
| Read/data strobe | RE_N | OE_N | read/data strobe | I | read control |
| Write/read or write | WE_N | WE_N | write/read or write | I | write control |
| INT | INT | INT | INT | O | interrupt request |
| DREQ | - | - | DREQ | O | DMA request |
| DACK | - | - | DACK | I | DMA acknowledge |

## 8.2 Interface mode lock

The bus interface can be locked in any of the modes, SRAM, NAND, NOR, or general multiplex, using bit 3 of the HW Mode Control register. To lock the interface in a particular mode:

1. Read bits 7 and 6 of the SW Reset register.
2. Set bit 3 of the HW Mode Control register to logic 1.
3. Read bits 7 and 6 of the SW Reset register to ensure that the interface is locked in the desired mode.

## 8.3 SRAM bus interface mode

The bus interface will be in SRAM 16-bit mode if pins ALE/ADV_N and CLE are HIGH, when:

- The CS_N/CE_N pin goes LOW, and the RD_N/DS_N/RE_N/OE_N pin goes LOW, or
- The RESET_N pin goes from LOW to HIGH.

Then, if the DATA_BUS_WIDTH bit is set, the bus interface will be in SRAM 8-bit mode.

In SRAM mode, A[7:0] is the 8-bit address bus and AD[15:0] is the separate 16-bit data bus. The ISP1763A pins RD_N/DS_N/RE_N/OE_N and WR_N/RW_N/WE_N are the read and write strobes. The SRAM bus interface supports both 8-bit and 16-bit bus width that can be configured by setting or clearing bit DATA_BUS_WIDTH. The DMA transfer is also applicable to this interface.

## 8.4 NAND bus interface mode

The bus interface will be in NAND 16-bit mode if pins ALE/ADV_N and CLE are LOW, when:

- The CS_N/CE_N pin goes LOW, and the RD_N/DS_N/RE_N/OE_N pin goes LOW, or
- The RESET_N pin goes from LOW to HIGH.

Then, if the DATA_BUS_WIDTH bit is set, the bus interface will be in NAND 8-bit mode.

The NAND bus interface supports most advance application processors. The command-address-data multiplexed bus is an 8-bit or 16-bit connection. The NAND Flash interface access consists of three phases: command, address, and data. The command phase is ignored. The address in the NAND Flash interface is sequentially sent in address cycles. For the ISP1763A application, an 8-bit address is sufficient to address all on-chip registers and buffers; see Figure 4. The last byte address latched will be the accessed address if there are several address cycles.

The data length can vary from one byte to multiple bytes. For example, to access the data port of the peripheral controller, the maximum data length can reach 1024 bytes.

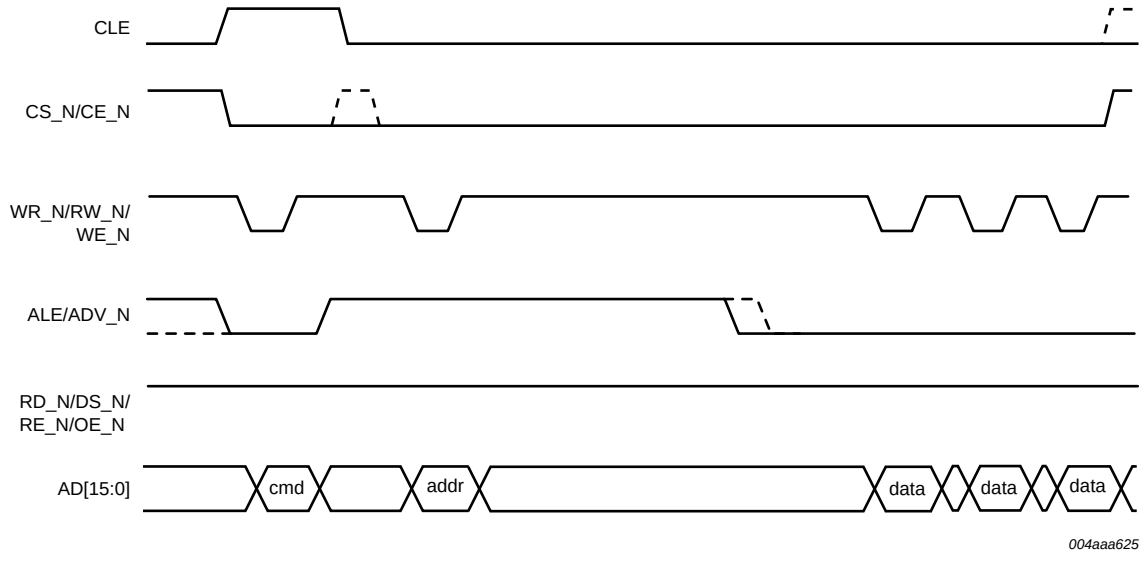



**Fig 4. Write operation**

## 8.5 NOR bus interface mode

The bus interface will be in NOR 16-bit mode, if pin ALE/ADV_N is HIGH and pin CLE is LOW, when:

- The CS_N/CE_N pin goes LOW, and the RD_N/DS_N/RE_N/OE_N pin goes LOW, or

- The RESET_N pin goes from LOW to HIGH.

Then if the DATA_BUS_WIDTH bit is set, the bus interface will be in NOR 8-bit mode.

The NOR Flash interface access consists of two phases: address and data.

The address is valid when CS_N/CE_N and ADV_N are LOW, and the address is latched at the rising edge of ADV_N. For a read operation, WE_N must be HIGH. OE_N is the data output control. When active, the addressed register or the buffer data is driven to the I/O bus. The read operation is completed when CS_N/CE_N is deasserted. For a write operation, OE_N must be HIGH. The WE_N assertion can start when ADV_N is deasserted. WE_N is the data input strobe signal. When deasserted, data will be written to the addressed register or the buffer. The write operation is completed when CS_N/CE_N is deasserted.

## 8.6 General multiplex bus interface mode

The bus interface will be in general multiplex 16-bit mode, if pin ALE/ADV_N is LOW and pin CLE is HIGH, when:

- The CS_N/CE_N pin goes LOW, and the RD_N/DS_N/RE_N/OE_N pin goes LOW, or
- The RESET_N pin goes from LOW to HIGH.

Then if the DATA_BUS_WIDTH bit is set, the bus interface will be in general multiplex 8-bit mode.The general multiplex bus interface supports most advance application processors. The general multiplex interface access consists of two phases: address and data.

The address is valid when ALE/ADV_N goes HIGH, and the address is latched at the falling edge of ALE/ADV_N. For a read operation, WR_N/RW_N/WE_N must be HIGH. RD_N/DS_N/RE_N/OE_N is the data output control. When active, the addressed register or the buffer data is driven to the I/O bus. The read operation is completed when CS_N/CE_N is deasserted. For a write operation, RD_N/DS_N/RE_N/OE_N must be HIGH. The WR_N/RW_N/WE_N assertion can start when ALE/ADV_N is deasserted. WR_N/RW_N/WE_N is the data input strobe signal. When deasserted, data will be written to the addressed register or the buffer. The write operation is completed when CS_N/CE_N is deasserted. The DMA transfer is also applicable to this interface.

## 8.7 DMA controller

The DMA controller of the ISP1763A is used to transfer data between the system memory and local buffers. It shares data bus AD[15:0] and control signals WR_N/RW_N/WE_N, RD_N/DS_N/RE_N/OE_N, and CS_N/CE_N. The logic is dependent on the bus interface mode setting.

DREQ is from the ISP1763A to indicate the start of DMA. DACK is used to differentiate if data transferred is for the DMA or PIO access. When DACK is asserted, it indicates that it is still in DMA mode. When DACK is deasserted, it indicates that PIO is to be accessed. ALE/ADV_N and CLE (address phase) are ignored in a DMA access cycle. Correct data will be captured only on the data phase (the rising edge of WR_N/RW_N/WE_N and RD_N/DS_N/RE_N/OE_N). The DMA controller of the ISP1763A has only one DMA channel. Therefore, only one DMA transfer may take place at a time.

The ISP1763A supports only counter mode. Dynamically assign the DMA transfer counter for each DMA transfer. The transfer ends once the transfer counter reaches zero. If the transfer counter is larger than the burst counter, the DREQ signal will deassert at the end of each burst transfer. DREQ will re-assert at the beginning of each burst.

For a 16-bit DMA transfer, the minimum burst length is 2 bytes. This means that the burst length is only one DMA cycle. Therefore, DREQ and DACK will assert and deassert at each DMA cycle.

In peripheral DMA, the bits in the Interrupt Reason register will be asserted to indicate that the DMA transfer has either successfully completed or terminated. Setting the control bits in the DMA Command register will start, stop, or reset the DMA transfer.

**Table 6. Register address**

| Device register | CPU address | Host register | CPU address | OTG register | CPU address |
|---|---|---|---|---|---|
| Address | 00h | USBCMD | 8Ch | OTG Control (set) | E4h |
| Mode | 0Ch | USBSTS | 90h | OTG Control (clear) | E6h |
| Interrupt Configuration | 10h | FRINDEX | 98h | OTG Status (RD only) | E8h |
| Debug | 12h | CONFIGFLAG | 9Ch | OTG Interrupt Latch (set) | ECh |
| DcInterruptEnable | 14h | PORTSC1 | A0h | OTG Interrupt Latch (clear) | EEh |
| Endpoint Index | 2Ch | ISO PTD Done Map | A4h | OTG Interrupt Enable Fall (set) | F0h |
| Control Function | 28h | ISO PTD Skip Map | A6h | OTG Interrupt Enable Fall (clear) | F2h |
| Data Port | 20h | ISO PTD Last PTD | A8h | OTG Interrupt Enable Rise (set) | F4h |
| Buffer Length | 1Ch | INT PTD Done Map | AAh | OTG Interrupt Enable Rise (clear) | F6h |
| DcBufferStatus | 1Eh | INT PTD Skip Map | ACh | OTG Timer (lower word: set) | F8h |
| EPMaxPacketSize | 04h | INT PTD Last PTD | AEh | OTG Timer (lower word: clear) | FAh |
| Endpoint Type | 08h | ATL PTD Done Map | B0h | OTG Timer (higher word: set) | FCh |
| DMA Command | 30h | ATL PTD Skip Map | B2h | OTG Timer (higher word: clear) | FEh |
| DMA Transfer Counter | 34h | ATL PTD Last PTD | B4h | | |
| DcDMAConfiguration | 38h | HW Mode Control | B6h | | |
| DMA Hardware | 3Ch | SW Reset | B8h | | |
| DMA Interrupt Reason | 50h | HcBufferStatus | BAh | | |
| DMA Interrupt Enable | 54h | HcDMAConfiguration | BCh | | |
| DMA Endpoint | 58h | ATL Done Timeout | C0h | | |
| DMA Burst Counter | 64h | Memory | C4h | | |
| DcInterrupt | 18h | Data | C6h | | |
| Chip ID | 70h | Edge Interrupt Count | C8h | | |
| Frame Number | 74h | DMA Start Address | CCh | | |
| Scratch | 78h | DMA Data Port | 60h | | |
| Unlock Device | 7Ch | Power Down Control | D0h | | |
| Interrupt Pulse Width | 80h | HcInterrupt | D4h | | |
| Test Mode | 84h | HcInterruptEnable | D6h | | |
| | | ISO IRQ MASK OR | D8h | | |
| | | INT IRQ MASK OR | DAh | | |
| | | ATL IRQ MASK OR | DCh | | |

**Table 6. Register address** *…continued*

| Device register | CPU address | Host register | CPU address | OTG register | CPU address |
|---|---|---|---|---|---|
| | | ISO IRQ MASK AND | DEh | | |
| | | INT IRQ MASK AND | E0h | | |
| | | ATL IRQ MASK AND | E2h | | |

## 8.8 On-The-Go (OTG) controller

### 8.8.1 Introduction

OTG is a supplement to the Hi-Speed USB specification that augments existing USB peripherals by adding to these peripherals limited host capability to support other targeted USB peripherals. It is primarily targeted at portable devices because it addresses concerns related to such devices, such as a small connector and low power. Non-portable devices, even standard hosts, can also benefit from OTG features.

The ISP1763A OTG controller is designed to perform all the tasks specified in the OTG supplement. It supports HNP and SRP for dual-role devices. The ISP1763A uses the software implementation of HNP and SRP for maximum flexibility. A set of OTG registers provides the control and status monitoring capabilities to support software HNP or SRP.

USB transceivers, timers, and analog components required by OTG are also integrated on-chip. The analog components include:

- Voltage comparators
- Pull-up or pull-down resistors on data lines
- Charging or discharging resistors for $V_{BUS}$

### 8.8.2 Dual-role device

When port 1 of the ISP1763A is configured in OTG mode, it can be used as an OTG dual-role device. A dual-role device is a USB device that can function either as a host or as a peripheral. As a host, the ISP1763A can support all four types of transfers, control, bulk, isochronous, and interrupt, at high-speed, full-speed, or low-speed. As a peripheral, the ISP1763A can support two control endpoints, and up to seven IN endpoints and seven OUT endpoints that can be programmed to any of the four transfer types.

The default role of the ISP1763A is controlled by the ID pin, which in turn is controlled by the type of plug connected to the micro-AB receptacle. If ID = LOW (micro-A plug connected), it becomes an A-device, which is a host by default. If ID = HIGH (micro-B plug connected), it becomes a B-device, which is a peripheral by default.

Both the A-device and the B-device work on a session basis. A session is defined as the period of time during which devices exchange data. A session starts when $V_{BUS}$ is driven and ends when $V_{BUS}$ is turned off. Both the A-device and the B-device may start a session. During a session, the role of the host can be transferred back and forth between the A-device and the B-device any number of times by using HNP.

If the A-device wants to start a session, it turns on $V_{BUS}$ by enabling the external charge pump. The B-device detects that $V_{BUS}$ has risen above the B_SESS_VLD level and assumes the role of a peripheral asserting its pull-up resistor on the DP line. The A-device detects the remote pull-up resistor and assumes the role of a host. Then the A-device can

communicate with the B-device as long as it wishes. When the A-device finishes communicating with the B-device, the A-device turns off $V_{BUS}$ and both the devices finally go into the idle state.

If the B-device wants to start a session, it must initiate SRP by data line pulsing and $V_{BUS}$ pulsing. When the A-device detects any of these SRP events, it turns on its $V_{BUS}$. (Note: only the A-device is allowed to drive $V_{BUS}$.) The B-device assumes the role of a peripheral, and the A-device assumes the role of a host. The A-device detects that the B-device can support HNP by getting the OTG descriptor from the B-device. The A-device will then enable the HNP hand-off by using SetFeature (b_hnp_enable) and then go into the suspend state. The B-device signals it is claiming the host role by deasserting its pull-up resistor. The A-device acknowledges by going into the peripheral state. The B-device then assumes the role of a host and communicates with the A-device as long as it wishes. When the B-device finishes communicating with the A-device, both the devices finally go into the idle state.

### 8.8.3 Session Request Protocol (SRP)

As a dual-role device, the ISP1763A can initiate and respond to SRP. The B-device initiates SRP by data line pulsing, followed by $V_{BUS}$ pulsing. The A-device can detect data line pulsing.

#### 8.8.3.1 B-device initiating SRP

The ISP1763A can initiate SRP by performing the following steps:

1. Detect initial conditions (read ID_GND, B_SESS_END, and SE0_2MS of the OTG Interrupt Source register).
2. Start data line pulsing (set DP_PULLUP of the OTG Control register to logic 1).
3. Wait for 5 ms to 10 ms.
4. Stop data line pulsing (set DP_PULLUP of the OTG Control register to logic 0).
5. Start $V_{BUS}$ pulsing (set VBUS_CHRG of the OTG Control register to logic 1).
6. Wait for 10 ms to 20 ms.
7. Stop $V_{BUS}$ pulsing (set VBUS_CHRG of the OTG Control register to logic 0).
8. Discharge $V_{BUS}$ for about 30 ms (by using VBUS_DISCHRG of the OTG Control register), optional.

The B-device must complete both data line pulsing and $V_{BUS}$ pulsing within 100 ms.

#### 8.8.3.2 A-device responding to SRP

The A-device must be able to respond to one of the two SRP events: data line pulsing or $V_{BUS}$ pulsing.

### 8.8.4 Host Negotiation Protocol (HNP)

HNP is used to transfer control of the host role between the default host (A-device) and the default peripheral (B-device) during a session. When the A-device is ready to give up its role as a host, it will condition the B-device using SetFeature (b_hnp_enable) and will go into suspend. If the B-device wants to use the bus at that time, it signals a disconnect to the A-device. Then, the A-device will take the role of peripheral and the B-device will take the role of a host.

#### 8.8.4.1 Sequence of HNP events

The sequence of events for HNP as observed on the USB bus is illustrated in Figure 5.

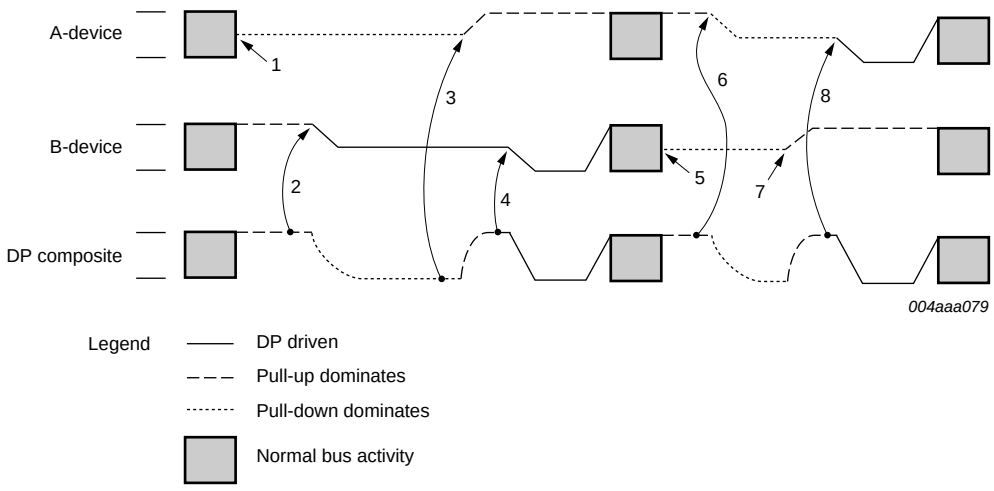



**Fig 5. HNP sequence of events**

As can be seen in Figure 5:

1. The A-device completes using the bus and stops all bus activity (that is, suspends the bus).
2. The B-device detects that the bus is idle for more than 5 ms and begins HNP by turning off the pull-up on DP. This allows the bus to discharge to the SE0 state.
3. The A-device detects SE0 on the bus and recognizes this as a request from the B-device to become a host. The A-device responds by turning on its DP pull-up within 3 ms of first detecting SE0 on the bus.
4. After waiting for 30 μs to ensure that the DP line is not HIGH because of the residual effect of the B-device pull-up, the B-device notices that the DP line is HIGH and the DM line is LOW, that is, J state. This indicates that the A-device has recognized the HNP request from the B-device. At this point, the B-device becomes a host and asserts bus reset to start using the bus. The B-device must assert the bus reset, that is, SE0, within 1 ms of the time that the A-device turns on its pull-up.
5. When the B-device completes using the bus, it stops all bus activities. Optionally, the B-device may turn on its DP pull-up at this time.
6. The A-device detects lack of bus activity for more than 3 ms and turns off its DP pull-up. Alternatively, if the A-device has no further need to communicate with the B-device, the A-device may turn off $V_{BUS}$ and end the session.
7. The B-device turns on its pull-up.
8. After waiting 30 μs to ensure that the DP line is not HIGH because of the residual effect of the A-device pull-up, the A-device notices that the DP line is HIGH (and the DM line is LOW) indicating that the B-device is signaling a connect and is ready to respond as a peripheral. At this point, the A-device becomes a host and asserts the bus reset to start using the bus.

### 8.8.5 Power saving in the idle state and during wake-up

The ISP1763A can be put in power saving mode if the OTG device is not in a session. This significantly reduces the power consumption. In this mode, both the peripheral controller and the host controller are suspended, the PLL and the oscillator are stopped, and the external charge pump is in the suspend state.

As an OTG device, however, the ISP1763A is required to respond to SRP events. To support this, a LazyClock is kept running when the chip is in power-saving mode. An SRP event will wake up the chip, that is, enable the PLL and the oscillator. Besides this, an ID change or B_SESS_VLD detection can also wake up the chip. These wake-up events can be enabled or disabled by programming the related bits of the OTG Interrupt Enable register before putting the chip in power saving mode. If the bit is set, then the corresponding event (status change) will wake up the ISP1763A. If the bit is cleared, then the corresponding event will not wake up the ISP1763A.

You can also wake up the ISP1763A from power-saving mode by using the software. This is accomplished by accessing any of the ISP1763A registers. Accessing a register will assert CS_N/CE_N and RD_N/DS_N/RE_N/OE_N of the ISP1763A, and therefore, set it awake.

## 8.9 USB host controller

### 8.9.1 ISP1763A USB host controller and hub internal architecture

The EHCI block and the Hi-Speed USB hub block are the main components of the advanced ST-Ericsson's slave host controller.

The EHCI is the latest generation design with improved data bandwidth. The EHCI in the ISP1763A is adapted from *Enhanced Host Controller Interface Specification for Universal Serial Bus Rev. 1.0*.

The internal Hi-Speed USB hub block replaces the companion host controller block used in the original architecture of a PCI Hi-Speed USB host controller to handle full-speed and low-speed modes. The host controller of the ISP1763A architecture is a simplified hardware architecture that helps reduce cost and development time by eliminating the additional work involved in implementing the OHCI software required to support full-speed and low-speed modes.

Figure 6 shows the internal architecture of the ISP1763A host controller. The ISP1763A host controller implements an EHCI that has one internal port, the root hub port 1 (not available externally), on which the internal hub is connected. The two external ports are always routed to the internal hub. The internal hub is a Hi-Speed USB, including the TT.

At power-on reset, followed by the host controller reset and initialization, the internal root hub port 1 will be polled until a new connection is detected showing the connection of the internal hub.

The internal Hi-Speed USB hub is enumerated using a sequence similar to a standard Hi-Speed USB hub enumeration sequence, and the polling on the root hub is stopped because the internal Hi-Speed USB hub will never be disconnected.

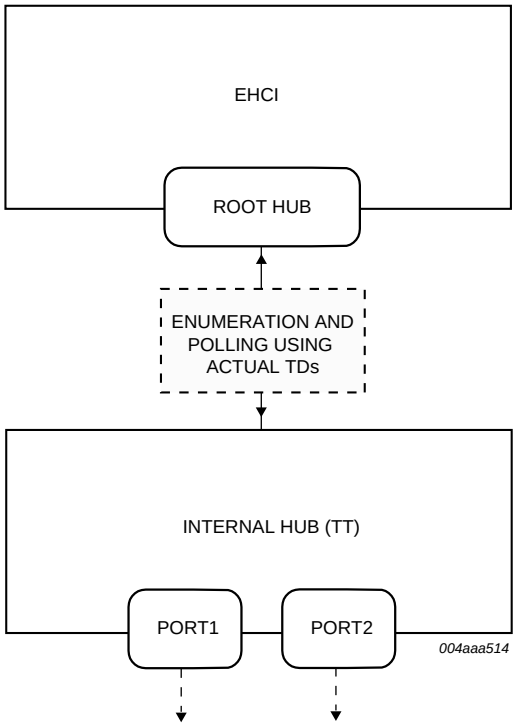



Fig 6. Internal hub

### 8.9.2 Host controller buffer memory block

#### 8.9.2.1 General considerations

The internal addressable host controller buffer memory is 24 kB.

The total amount of memory allocated to the payload determines the maximum transfer size specified by a PTD, a larger internal memory size results in lesser CPU interruption for transfer programming. This means less time spent in context switching, resulting in better CPU usage.

A larger buffer also implies a larger amount of data to be transferred. This transfer, however, can be done over a longer period of time, to maintain overall system performance. Each transfer of the USB data on the USB bus can span up to a few milliseconds before requiring further CPU intervention for data movement.

The internal architecture of the ISP1763A host controller allows a flexible definition of the memory buffer for optimization of the data transfer on the CPU extension bus and the USB. It is possible to implement different data transfer schemes, depending on the number and type of USB devices present. For example, push-pull: data can be written to half of the memory while data in the other half is accessed by the host controller and sent on the USB bus. This is useful especially when a high-bandwidth continuous or periodic data flow is required.

#### 8.9.2.2 Structure of the host controller memory

The internal memory is 24 kB: 4 kB PTD area and 20 kB payload area.

Both the PTD and payload memory zones are divided into three dedicated areas for each main type of USB transfer: Isochronous (ISO), Interrupt (INT), and Asynchronous Transfer List (ATL). As shown in Table 7, the PTD areas for ISO, INT, and ATL are grouped at the beginning of the memory, occupying address range 0400h to 0FFFh, following the registers address space. The payload or data area occupies the next memory address range 1000h to 5FFFh, meaning that 20 kB of memory is allocated for the payload data.

A maximum of 16 PTD areas and their allocated payload areas can be defined for each type of transfer. The structure of a PTD is similar for every transfer type and consists of eight Double Words (DWs) that must be correctly programmed for correct USB data transfer. The reserved bits of a PTD must be set to logic 0. A detailed description of the PTD structure can be found in Section 10.4.

The transfer size specified by the PTD determines the contiguous USB data transfer that can be performed without any CPU intervention. The respective payload memory area must be equal to the transfer size defined. The maximum transfer size is flexible and can be optimized, depending on the number and nature of USB devices or PTDs defined and their respective MaxPacketSize.

The RAM is structured in blocks of PTDs and payloads so that while the USB is executing on an active transfer-based PTD, the processor can simultaneously fill up another block area in the RAM. A PTD and its payload can then be updated on-the-fly without stopping or delaying any other USB transaction or corrupting the RAM data.

Some of the design features are:

- The internal memory contains isochronous, interrupt, and asynchronous PTDs, and defined payloads.
- Internal memory address range calculation:

  Memory address = (CPU address − 0400h) (shift right >> 3). The base address is 0400h.

**Table 7. Memory address**

| Memory map | CPU address | Memory address | Memory accessing |
|---|---|---|---|
| ISO PTD | 0400h to 05FFh | 0000h to 007Fh | • Configure the Memory register<br>– Bits 14 to 0: start address 1000h to 5FFFh<br>– Bit 15: reserved<br>• Read/write data from or to the Data register (address C6h of the ISP1763A).<br>• The memory burst read and write is ended by any register access (other than C6h). |
| INT PTD | 0800h to 09FFh | 0080h to 00FFh | |
| ATL PTD | 0C00h to 0DFFh | 0100h to 017Fh | |
| Payload | 1000h to 5FFFh | 0180h to 0B7Fh | |

Both the CPU interface logic and the USB host controller require access to the internal ISP1763A RAM at the same time. The internal arbiter controls these accesses to the internal memory, organized internally on a 64-bit data bus width, allowing a maximum bandwidth of 240 MB/s. This bandwidth avoids any bottleneck on accesses both from the CPU interface and the internal USB host controller.

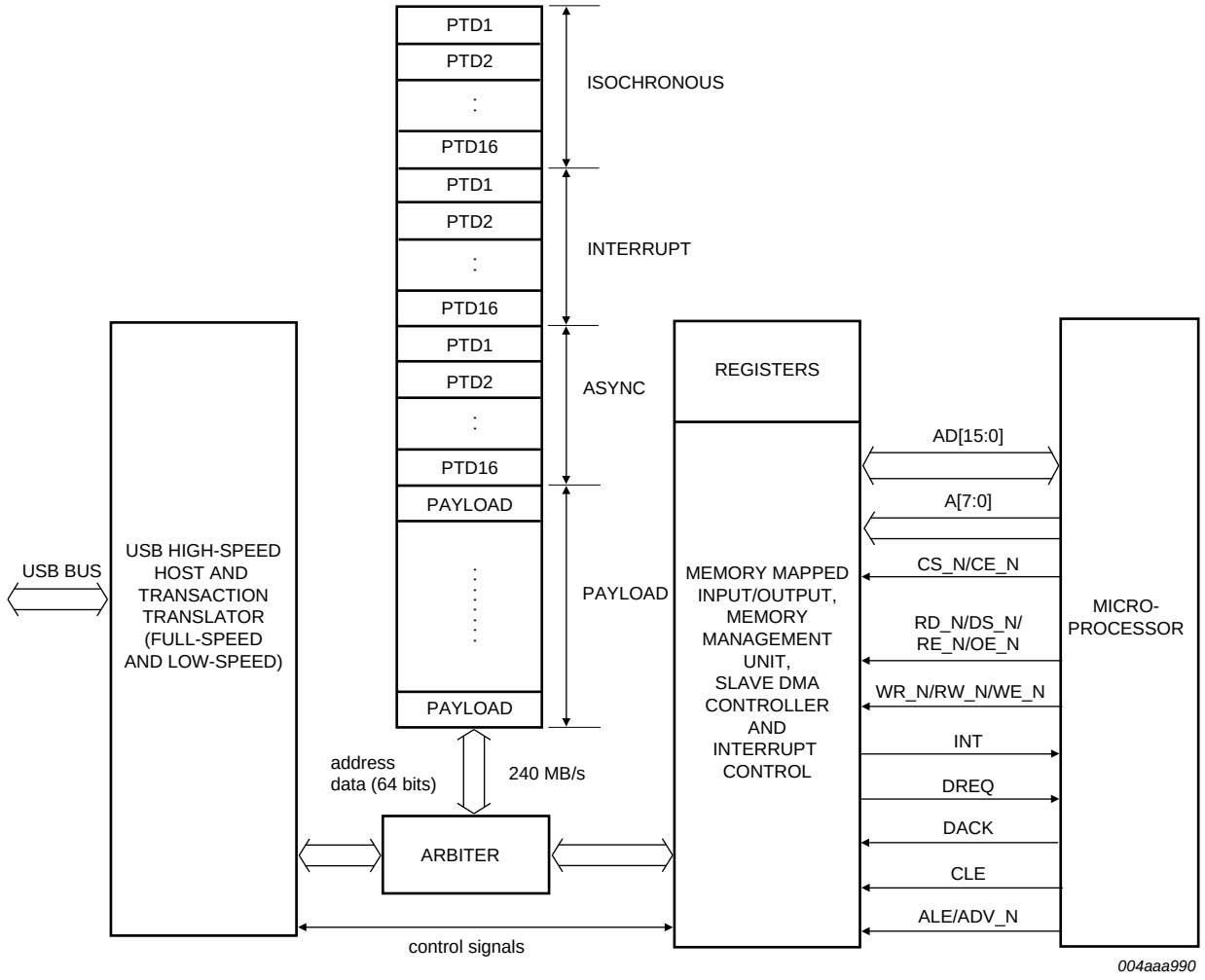



**Fig 7. Memory segmentation and access block diagram**

### 8.9.3 Interrupts

The ISP1763A will generate an INT according to the source or event in the HcInterrupt register. The main steps to enable the INT assertion are:

1. Set GLOBAL_INTR_EN (bit 0) in the HW Mode Control register.
2. Define the INT active as level or edge in INTR_LEVEL (bit 1) of the HW Mode Control register.
3. Define the INT polarity as active LOW or HIGH in INTR_POL (bit 2) of the HW Mode Control register. These settings must match the INT settings of the host processor.

   By default, interrupt is level-triggered and active LOW.
4. Program the individual interrupt enable bits in the HcInterruptEnable register. The software will need to clear the interrupt status bits in the HcInterrupt register before enabling individual interrupt enable bits.

Additional INT characteristics can be adjusted in the Edge Interrupt Count register, as necessary, applicable only when INT is set to be edge-active; a pulse of a defined width is generated every time INT is active.

Bits 15 to 0 of the Edge Interrupt Count register will define the INT pulse width. The maximum pulse width that can be programmed is FFFFh, corresponding to a 1 ms pulse width. This setting is necessary for certain processors that may require a different minimum INT pulse width than the default value. The default INT width set at power-on is about 500 ns.

Bits 31 to 24 of the Edge Interrupt Count register define the minimum interval between two interrupts to avoid frequent INT received by the CPU. The default value of 00h attributed to these bits determines the normal INT generation, without any delay. When a delay is programmed and the INT becomes active after that delay, several INT events may already have occurred.

All the interrupt events are represented by the respective bits allocated in the HcInterrupt register. There is no mechanism to show the order or the moment occurrence of an interrupt.

The asserted bits in the HcInterrupt register can be cleared by writing back the same value to the HcInterrupt register. This means that writing logic 1 to each of the set bits will reset those bits to the initial inactive state.

The INT generation rules that apply according to the preceding settings are:

- If an interrupt event occurs but the corresponding bit in the HcInterruptEnable register is not set, then the corresponding HcInterrupt register bit is set but the interrupt signal is not asserted.

  An interrupt will be generated when interrupt is enabled and the source is set.
- For a level-trigger, an interrupt signal remains asserted until the processor clears the HcInterrupt register by writing logic 1 to clear the HcInterrupt register bits that are set.
- If an interrupt is made edge-sensitive and is asserted, writing to clear the HcInterrupt register will not have any effect because the interrupt will be asserted for a prescribed number of clock cycles.
- The clock stopping mechanism does not affect the generation of an interrupt. This is useful during the suspend and resume cycles, when an interrupt is generated to signal a wake-up event.

The INT generation can also be conditioned by programming the IRQ MASK OR and IRQ MASK AND registers.

With the help of the IRQ MASK AND and IRQ MASK OR registers for each type of transfer, the software can determine which PTDs get priority and an interrupt will be generated when the AND or OR conditions are met. The PTDs that are set will wait until the respective bits of the remaining PTDs are set and then all PTDs generate an interrupt request to the CPU together.

The registers definition shows that the AND or OR conditions are applicable to the same category of PTDs: ISO, INT, and ATL.

When an INT is generated, the PTD Done Map registers and the respective V bits will show which PTDs were completed.

The rules that apply to IRQ MASK AND or IRQ MASK OR settings are:

- The OR mask has a higher priority over the AND mask. An INT is generated if bit n of the done map is set and the corresponding bit n of the OR MASK register is set.
- If the OR mask for any done bit is not set, then the AND mask comes into picture. An INT is generated if all the corresponding done bits of the AND MASK register are set. For example: If bits 2, 4, and 10 are set in the AND MASK register, an INT is generated only if bits 2, 4, and 10 of done map are set.
- If using the INT interval setting for the bulk PTD, an interrupt will only occur at the regular time interval as programmed in the ATL Done Timeout register.

  Even if an interrupt occurs before the time-out of the register, no INT will be generated until the time is up.

**Example**: Using IRQ MASK AND or IRQ MASK OR, without ATL Timeout register.

The AND function: activate the INT only if PTDs 1, 2, and 4 are done.

The OR function: if any of the PTDs 7, 8, or 9 are done, an INT for each of the PTDs will be raised.

**Table 8. Using the IRQ MASK AND or IRQ MASK OR registers**

| PTD | AND register | OR register | Time | PTD done | INT |
|---|---|---|---|---|---|
| 1 | 1 | 0 | 1 ms | 1 | - |
| 2 | 1 | 0 | - | 1 | - |
| 3 | 0 | 0 | - | - | - |
| 4 | 1 | 0 | 3 ms | 1 | active because of AND |
| 5 | 0 | 0 | - | - | - |
| 6 | 0 | 0 | - | - | - |
| 7 | 0 | 1 | 5 ms | 1 | active because of OR |
| 8 | 0 | 1 | 6 ms | 1 | active because of OR |
| 9 | 0 | 1 | 7 ms | 1 | active because of OR |

## 8.10 USB peripheral controller

### 8.10.1 Introduction

The design of the peripheral controller in the ISP1763A is compatible with the ST-Ericsson's *ISP1582 Hi-Speed Universal Serial Bus peripheral controller* IC. The functionality of the peripheral controller in the ISP1763A is similar to the ISP1582. In addition, the register sets are similar, with only a few variations.

The USB Chapter 9 protocol handling and data transfer operation of the peripheral controller are executed using an external firmware. The external microcontroller or microprocessor can access the peripheral controller-specific registers through the local bus interface. The transfer of data between a microprocessor and the peripheral controller can be done in PIO mode.

### 8.10.2 Peripheral controller data transfer operation

The following sections explain how the peripheral controller in the ISP1763A handles an IN data transfer and an OUT data transfer. An IN data transfer means transfer from the ISP1763A to an external USB host, through the upstream port. An OUT transfer means transfer from an external USB host to the ISP1763A. In peripheral mode, the ISP1763A acts as a USB peripheral.

#### 8.10.2.1 IN data transfer

- The arrival of the IN token is detected by the Serial Interface Engine (SIE) by decoding the Packet Identifier (PID).
- The SIE also checks the device number and the endpoint number to verify whether they are okay.
- If the endpoint is enabled, the SIE checks the endpoint status. If the endpoint is full and data in the buffer is validated, the contents of the buffer memory are sent during the data phase; else an NAK handshake is sent.
- After the data phase, the SIE expects a handshake (ACK) from the host, except for ISO endpoints.
- On receiving the handshake (ACK), the SIE updates the contents of the endpoint status and interrupt registers, which in turn generates an interrupt to the microprocessor. For ISO endpoints, the DcInterrupt register is updated as soon as data is sent because there is no handshake phase.
- On receiving an interrupt, the microprocessor reads the DcInterrupt register. It knows which endpoint has generated the interrupt. If the buffer is empty, it fills up the buffer so that data can be sent by the SIE at the next IN token phase.

#### 8.10.2.2 OUT data transfer

- The arrival of the OUT token is detected by the SIE by decoding the PID.
- The SIE checks the device and endpoint numbers to verify whether they are okay.
- If the endpoint is enabled, the SIE checks the status of the endpoint. If the endpoint is empty, data from the USB host is stored in the buffer memory during the data phase, else a NAK handshake is sent.
- After the data phase, the SIE sends a handshake (ACK) to the host, except for ISO endpoints.
- The SIE updates the endpoint status and interrupt registers, which in turn generates an interrupt to the microprocessor. For ISO endpoints, the DcInterrupt register is updated as soon as data is received because there is no handshake phase.
- On receiving an interrupt, the microprocessor reads the DcInterrupt register. It knows which endpoint has generated the interrupt. If the buffer is full, it empties the buffer so that data can be received by the SIE at the next OUT token phase.

### 8.10.3 Endpoint description

Each USB peripheral is logically composed of several independent endpoints. An endpoint acts as a terminus of a communication flow between the USB host and the USB device. At design time, each endpoint is assigned a unique endpoint identifier, see

Table 9. The combination of the peripheral address (given by the host during enumeration), the endpoint number, and the transfer direction allows each endpoint to be uniquely referenced.

The peripheral controller has 4 kB of internal FIFO memory, which is shared among the enabled USB endpoints. The two control endpoints are fixed 64 bytes long. Any of the seven IN and seven OUT endpoints can separately be enabled or disabled. The endpoint type (interrupt, isochronous, or bulk) and packet size of these endpoints can individually be configured, depending on the requirements of the application. Optional double buffering increases the data throughput of these data endpoints.

**Table 9. Endpoint access and programmability**

| Endpoint identifier | Maximum packet size | Double buffering | Endpoint type | Direction |
|---|---|---|---|---|
| EP0RX | 64 bytes (fixed) | no | control OUT | OUT |
| EP0TX | 64 bytes (fixed) | no | control IN | IN |
| EP1RX | programmable | yes | programmable | OUT |
| EP1TX | programmable | yes | programmable | IN |
| EP2RX | programmable | yes | programmable | OUT |
| EP2TX | programmable | yes | programmable | IN |
| EP3RX | programmable | yes | programmable | OUT |
| EP3TX | programmable | yes | programmable | IN |
| EP4RX | programmable | yes | programmable | OUT |
| EP4TX | programmable | yes | programmable | IN |
| EP5RX | programmable | yes | programmable | OUT |
| EP5TX | programmable | yes | programmable | IN |
| EP6RX | programmable | yes | programmable | OUT |
| EP6TX | programmable | yes | programmable | IN |
| EP7RX | programmable | yes | programmable | OUT |
| EP7TX | programmable | yes | programmable | IN |

### 8.10.4 Peripheral controller suspend

The peripheral controller in the ISP1763A detects a USB suspend when constant idle state is present on the USB bus for 3 ms.

The steps leading the peripheral controller to the suspend state are as follows:

1. If there is no SOF for 3 ms, the peripheral controller in the ISP1763A sets bit SUSP of the DcInterrupt register. This will generate an interrupt if bit IESUSP of the DcInterruptEnable register is set.
2. When the firmware detects a suspend condition through bit IESUSP, it must prepare all system components for the suspend state.
3. In the interrupt service routine, the firmware must check the current status of the USB bus. When bit VBUSSTAT of the Mode register is logic 0, the USB bus has left suspend mode and the process must be aborted. Otherwise, the next step can be executed.
4. To meet the suspend current requirements for a bus-powered device, internal clocks must be switched off by clearing bit CLKAON of the Mode register.

5. When the firmware has set and cleared the GOSUSP bit of the Mode register, the peripheral controller in the ISP1763A enters the suspend state. A flag must be set by the firmware to indicate that the peripheral controller is in the suspend state.

The peripheral controller in the ISP1763A will remain in the suspend state for at least 5 ms, before responding to wake-up events, such as global resume or chip select active.

### 8.10.5 Peripheral controller resume

Wake-up from the suspend state is initiated either by the USB host or by the application:

- USB host: drives a K-state on the USB bus (global resume).
- Application: remote wake-up using a LOW pulse on pins CS_N/CE_N and RD_N/DS_N/RE_N/OE_N, if enabled using bit WKUPCS of the Mode register.

The steps of a wake-up sequence are as follows:

1. The internal oscillator and the PLL multiplier are re-enabled. When stabilized, clock signals are routed to all internal circuits of the peripheral controller in the ISP1763A.
2. The RESUME bit of the DcInterrupt register is set. This will generate an interrupt if bit IERESM of the DcInterruptEnable register is set.
3. The peripheral controller in the ISP1763A resumes its normal functionality 5 ms after starting the wake-up sequence. The firmware can clear its suspend state flag at this point.
4. After resume, the internal registers of the peripheral controller in the ISP1763A are write-protected to prevent corruption by inadvertent writing during power-up of external components. The firmware must send an Unlock Device command to the peripheral controller in the ISP1763A to restore its full functionality.

### 8.10.6 Remote wake-up

In a remote wake-up to the host, the firmware must set and clear the SNDRSU bit of the Mode register. The peripheral controller in the ISP1763A will drive a resume signal (a K-state) on the USB bus for 10 ms after a 5 ms delay.

## 8.11 Phase-Locked Loop (PLL) clock multiplier

The internal PLL supports 12 MHz, 19.2 MHz, or 24 MHz input, which can be a crystal or a clock already existing in the system. The frequency selection can be done using the FREQSEL1 and FREQSEL2 pins.

No external components are required for the PLL operation.

## 8.12 Power management

The ISP1763A is mainly designed for mobile applications that require more precision power saving control to achieve extremely low power consumption. It implements a flexible power management scheme that allows various stages of power saving.

### 8.12.1 Power supply

Power supplies are defined in Table 10.

Table 10. Power supply

| Symbol | Voltage range | Description |
|---|---|---|
| $V_{CC(I/O)}$ | 1.65 V to 1.95 V or 3.0 V to 3.6 V | supply for the I/O pad |
| $V_{CC(3V3)}$ | 3.0 V to 3.6 V | supply for USB transceivers, cores, and analog modules |

### 8.12.2 Power modes

#### 8.12.2.1 Operation mode

All power supplies are present. Consists of host mode, peripheral mode, and idle mode.

#### 8.12.2.2 Suspend mode

All power supplies are present. Possible defined states are host-only suspend, peripheral-only suspend, or both.

For the peripheral suspend procedure, see Section 8.10.4.

The steps for the host suspend are as follows:

1. Clear the RS bit of the USBCMD register to stop the host controller from executing schedule.
2. Set the SUSP bit and clear the FPR bit of the PORTSC1 register to force the host controller to go into suspend.

#### 8.12.2.3 Deep sleep mode

All power supplies are present. Regulator is in suspend mode. The clocks of the host controller and the peripheral controller are turned off.

The steps to enter deep sleep mode are:

1. The peripheral must be in suspend state or disabled. See Section 8.10.4.
2. Clear the RS bit of the USBCMD register to stop the host controller from executing schedule.
3. Clear the CLKAON bit in the Mode register to save power.
4. Set the REG_PWR and REG_SUSP_PWR bits of the Power Down Control register to logic 1 to force the regulator to go into suspend mode.

#### 8.12.2.4 Power-down mode

The regulator is powered down. The ISP1763A has no functionality. The ISP1763A can be woken up by a dummy read signal, that is, both RD_N/DS_N/RE_N/OE_N and CS_N/CE_N are active LOW.

The ISP1763A enters power-down mode when any of the following conditions is met:

- Bit REG_PWR of the Power Down Control register is set to logic 0.
- RESET_N is asserted.

#### 8.12.2.5 ISP1763A wake up

The regulator will be in normal operating mode and the clock will be enabled when either of these conditions are triggered:

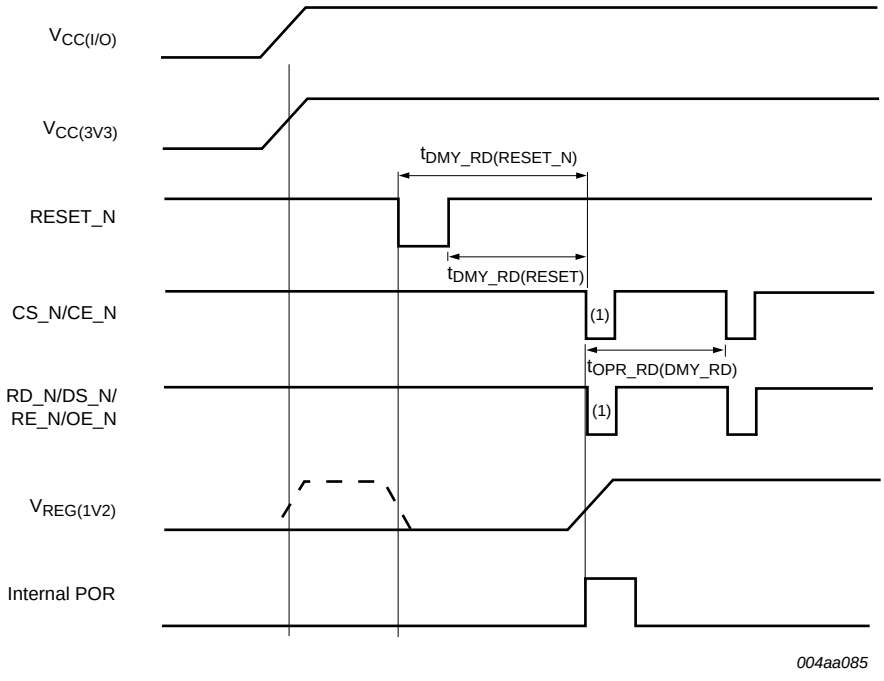
- Application: Dummy read access with a LOW pulse on pins CS_N/CE_N and RD_N/DS_N/RE_N/OE_N.
- Host: Remote wake up from the external USB device.
- Host: $V_{BUS}$ overcurrent condition triggered on the system.
- Peripheral: Resume signaling received from the external USB host.

#### 8.12.2.6 Isolation mode

All power supplies are not present. Although $V_{BUS}$ may be present, it will not activate the chip or damage it.

### 8.12.3 Power-up and reset sequence

When $V_{CC(I/O)}$ and $V_{CC(3V3)}$ are on, it is recommended that the system generates a RESET_N pulse to ensure that the ISP1763A regulator is in the power-down state. The regulator will be powered on when the system generates a dummy read (ignore return value) access to the ISP1763A. An internal POR pulse will be generated during the regulator powers on, so that internal circuits are in reset state after the regulator power is stable.




(1) Dummy read.

**Fig 8. Power-up and reset sequence**

### 8.12.4 ATX reference voltage

The ATX circuit provides a stable internal voltage reference to bias the analog circuitry. This circuit requires an accurate external reference resistor. Connect 12 kΩ ± 1% resistor between pins RREF1, RREF2, and GND.

# 9. OTG controller-specific registers

**Table 11. Overview of OTG controller-specific registers**

| Address | Register | Reset value | Access[1] | Reference |
|---|---|---|---|---|
| **OTG control register** | | | | |
| **Set —** E4h<br>**Clear —** E6h | OTG Control | 8086h | R/S/C | Section 9.1.1 on page 30 |
| **OTG interrupt registers** | | | | |
| E8h | OTG Interrupt Source | 1188h | R | Section 9.2.1 on page 31 |
| **Set —** ECh<br>**Clear —** EEh | OTG Interrupt Latch | 0000h | R/S/C | Section 9.2.2 on page 32 |
| **Set —** F0h<br>**Clear —** F2h | OTG Interrupt Enable Fall | 0000h | R/S/C | Section 9.2.3 on page 33 |
| **Set —** F4h<br>**Clear —** F6h | OTG Interrupt Enable Rise | 0000h | R/S/C | Section 9.2.4 on page 34 |
| **OTG timer register** | | | | |
| **Timer_Low_Word_Set —** F8h<br>**Timer_Low_Word_Clear —** FAh<br>**Timer_High_Word_Set —** FCh<br>**Timer_High_Word_Clear —** FEh | OTG Timer | 0000h | R/S/C | Section 9.3.1 on page 35 |

[1] The R/S/C access type represents a field that can be read, set, or cleared (set to 0). A set register is used to configure the function that is defined in the bit field of the register. The clear register is used to clear the configuration setting. Logic 1 in a bit field clears the function that is defined by the bit in the set register.

## 9.1 OTG control register

### 9.1.1 OTG Control register

Table 12 shows the bit allocation of the register.

**Table 12. OTG_CTRL - OTG Control register (address set: E4h, clear: E6h) bit allocation**

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| **Symbol** | HC_2_DIS | reserved | TMR_SEL | reserved | | OTG_ DISABLE | OTG_SE0_ EN | BDIS_ ACON_EN |
| **Reset** | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **Access** | R/S/C | R/S/C | R/S/C | R/S/C | R/S/C | R/S/C | R/S/C | R/S/C |
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| **Symbol** | SW_SEL_ HC_DC | VBUS_ CHRG | VBUS_ DISCHRG | VBUS_ DRV | reserved | DM_PULL DOWN | DP_PULL DOWN | DP_PULL UP |
| **Reset** | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 0 |
| **Access** | R/S/C | R/S/C | R/S/C | R/S/C | R/S/C | R/S/C | R/S/C | R/S/C |

Table 13. OTG_CTRL - OTG Control register (address set: E4h, clear: E6h) bit description

| Bit | Symbol | Description |
|---|---|---|
| 15 | HC_2_DIS | **0** — Port 2 is enabled as host.<br>**1** — Port 2 is disabled. |
| 14 | - | reserved |
| 13 | TMR_SEL | Port 1 peripheral suspend timer select<br>**0** — 3.1 ms<br>**1** — 3.0 ms |
| 12 to 11 | - | reserved |
| 10 | OTG_DISABLE | **0** — OTG functionality is enabled.<br>**1** — OTG disabled; pure host or peripheral. |
| 9 | OTG_SE0_EN | This bit is used by the host controller to send SE0 on remote connect.<br>**0** — No SE0 sent on remote connect detection.<br>**1** — SE0 (bus reset) sent on remote connect detection.<br>**Remark:** This bit is normally set when the B-device goes into the B_WAIT_ACON state (recommended sequence: LOC_CONN = 0 → DELAY → 0 ms → OTG_SE0_EN = 1 → SEL_HC_DC = 0) and is cleared when it comes out of the B_WAIT_ACON state. |
| 8 | BDIS_ACON_EN | Enables the A-device to connect if the B-device disconnect is detected. |
| 7 | SW_SEL_HC_DC | In software HNP mode, this bit selects between the host controller and the peripheral controller.<br>**0** — Host controller is connected to ATX.<br>**1** — Peripheral controller is connected to ATX.<br>This bit is set to logic 1 by the hardware when there is an event corresponding to the BDIS_ACON interrupt. BDIS_ACON_EN is set and there is an automatic pull-up connection on remote disconnect. |
| 6 | VBUS_CHRG | Connect $V_{BUS}$ to $V_{CC(3V3)}$ through a resistor. |
| 5 | VBUS_DISCHRG | Discharge $V_{BUS}$ to ground through a resistor. |
| 4 | VBUS_DRV | In OTG mode, setting this bit will turn on port 1 power. In non-OTG mode, setting this bit does not have any effect and $V_{BUS}$ is controlled using the hub command. |
| 3 | - | reserved |
| 2 | DM_PULLDOWN | **0** — Disable DM pull-down<br>**1** — Enable DM pull-down |
| 1 | DP_PULLDOWN | **0** — Disable DP pull-down<br>**1** — Enable DP pull-down |
| 0 | DP_PULLUP | **0** — The pull-up resistor is disconnected from the DP line.<br>**1** — An internal 1.5 kΩ pull-up resistor is present on the DP line. |

## 9.2 OTG interrupt registers

### 9.2.1 OTG Interrupt Source register

This register indicates the current state of the signals that can generate an interrupt. The bit allocation of the register is given in Table 14.

**Table 14. OTG_INTR_SRC - OTG Interrupt Source register (address E8h) bit allocation**

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Symbol | reserved | | | | DP2_SRP | P2_A_SESS_VLD | OTG_TMR_TIMEOUT | B_SE0_SRP |
| Reset | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 |
| Access | R | R | R | R | R | R | R | R |
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| Symbol | B_SESS_END | reserved | | RMT_CONN | ID | DP_SRP | A_B_SESS_VLD | VBUS_VLD |
| Reset | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| Access | R | R | R | R | R | R | R | R |

**Table 15. OTG_INTR_SRC - OTG Interrupt Source register (address E8h) bit description**

| Bit | Symbol | Description |
|---|---|---|
| 15 to 12 | - | reserved |
| 11 | DP2_SRP | Detects the DP2 pin status. |
| 10 | P2_A_SESS_VLD | Reflects the voltage on pin OC2/$V_{BUS2}$ is above the A-device session valid threshold. |
| 9 | OTG_TMR_TIMEOUT | Reflects the OTG timer timeout. |
| 8 | B_SE0_SRP | Detects 2 ms of SE0 in the B-device idle state. |
| 7 | B_SESS_END | Reflects the voltage on pin OC1/$V_{BUS1}$ is below the B-device session end threshold. |
| 6 to 5 | - | reserved |
| 4 | RMT_CONN | Reflects remote connection is detected. |
| 3 | ID | Reflects the ID pin status. |
| 2 | DP_SRP | Detects the DP1 pin status. |
| 1 | A_B_SESS_VLD | Reflects the voltage on pin OC1/$V_{BUS1}$ is above the A-device session valid threshold for the A-device. Reflects the voltage on pin OC1/$V_{BUS1}$ is above the B-device session valid threshold for the B-device. |
| 0 | VBUS_VLD | Reflects the voltage on pin OC1/$V_{BUS1}$ is above the A-device $V_{BUS}$ valid threshold. |

### 9.2.2 OTG Interrupt Latch register

The OTG Interrupt Latch register indicates the source that generated the interrupt. The status of this register bits depends on the settings of the Interrupt Enable Fall and Interrupt Enable Rise registers, and the occurrence of the respective events. The bit allocation of the register is given in Table 16.

**Table 16. OTG_INTR_L - OTG Interrupt Latch register (address set: ECh, clear: EEh) bit allocation**

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Symbol | reserved | | | | DP2_SRP_L | P2_A_SESS_VLD_L | OTG_TMR_TIMEOUT_L | B_SE0_SRP_L |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/S/C | R/S/C | R/S/C | R/S/C | R/S/C | R/S/C | R/S/C | R/S/C |
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| Symbol | B_SESS_END_L | BDIS_ACON_L | reserved | RMT_CONN_L | ID_L | DP_SRP_L | A_B_SESS_VLD_L | VBUS_VLD_L |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/S/C | R/S/C | R/S/C | R/S/C | R/S/C | R/S/C | R/S/C | R/S/C |

Table 17. OTG_INTR_L - OTG Interrupt Latch register (address set: ECh, clear: EEh) bit description

| Bit | Symbol | Description |
|---|---|---|
| 15 to 12 | - | reserved |
| 11 | DP2_SRP_L | Set when an unmasked event occurs on DP2_SRP. |
| 10 | P2_A_SESS_VLD_L | Set when an unmasked event occurs on P2_A_SESS_VLD. |
| 9 | OTG_TMR_TIMEOUT_L | Set when an unmasked event occurs on OTG_TMR_TIMEOUT. |
| 8 | B_SE0_SRP_L | Set when an unmasked event occurs on B_SE0_SRP. |
| 7 | B_SESS_END_L | Set when an unmasked event occurs on B_SESS_END. |
| 6 | BDIS_ACON_L | Set when an unmasked event occurs on BDIS_ACON. |
| 5 | - | reserved |
| 4 | RMT_CONN_L | Set when an unmasked event occurs on RMT_CONN. |
| 3 | ID_L | Set when an unmasked event occurs on ID. |
| 2 | DP_SRP_L | Set when an unmasked event occurs on DP_SRP. |
| 1 | A_B_SESS_VLD_L | Set when an unmasked event occurs on A_B_SESS_VLD. |
| 0 | VBUS_VLD_L | Set when an unmasked event occurs on VBUS_VLD. |

### 9.2.3 OTG Interrupt Enable Fall register

Table 18 shows the bit allocation of this register that enables interrupts on transition from logic 1 to logic 0.

Table 18. OTG_INTR_EN_F - OTG Interrupt Enable Fall register (address set: F0h, clear: F2h) bit allocation

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Symbol | reserved | | | | DP2_SRP_EN_F | P2_A_SESS_VLD_EN_F | OTG_TMR_TIMEOUT_EN_F | B_SE0_SRP_EN_F |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/S/C | R/S/C | R/S/C | R/S/C | R/S/C | R/S/C | R/S/C | R/S/C |
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| Symbol | B_SESS_END_EN_F | BDIS_ACON_EN_F | reserved | RMT_CONN_EN_F | ID_EN_F | DP_SRP_EN_F | A_B_SESS_VLD_EN_F | VBUS_VLD_EN_F |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/S/C | R/S/C | R/S/C | R/S/C | R/S/C | R/S/C | R/S/C | R/S/C |

Table 19. OTG_INTR_EN_F - OTG Interrupt Enable Fall register (address set: F0h, clear: F2h) bit description

| Bit | Symbol | Description |
|---|---|---|
| 15 to 12 | - | reserved |
| 11 | DP2_SRP_EN_F | Enable interrupt when port 2 DP transitions from logic 1 to logic 0. |
| 10 | P2_A_SESS_VLD_EN_F | Enable interrupt when port 2 A_SESS_VLD transitions from logic 1 to logic 0 for the A-device. |
| 9 | OTG_TMR_TIMEOUT_EN_F | Enable interrupt on OTG timer time-out transition from logic 1 to logic 0. |
| 8 | B_SE0_SRP_EN_F | Enable interrupt when B_SE0_SRP transitions from logic 1 to logic 0. |
| 7 | B_SESS_END_EN_F | Enable interrupt when $V_{BUS}$ session end transitions from logic 1 to logic 0. |
| 6 | BDIS_ACON_EN_F | Enable interrupt when BDIS_ACON transitions from logic 1 to logic 0. |
| 5 | - | reserved |

Table 19. OTG_INTR_EN_F - OTG Interrupt Enable Fall register (address set: F0h, clear: F2h) bit description

| Bit | Symbol | Description |
|---|---|---|
| 4 | RMT_CONN_EN_F | Enable interrupt when RMT_CONN transitions from logic 1 to logic 0. |
| 3 | ID_EN_F | Enable interrupt when ID transitions from logic 1 to logic 0. |
| 2 | DP_SRP_EN_F | Enable interrupt when port 1 DP transitions from logic 1 to logic 0. |
| 1 | A_B_SESS_VLD_EN_F | Enable interrupt when AB-session valid transitions from logic 1 to logic 0. |
| 0 | VBUS_VLD_EN_F | Enable interrupt when VBUS_VLD transitions from logic 1 to logic 0. |

### 9.2.4 OTG Interrupt Enable Rise register

This register (see Table 20 for bit allocation) enables interrupts on transition from logic 0 to logic 1.

Table 20. OTG_INTR_EN_R - OTG Interrupt Enable Rise register (address set: F4h, clear: F6h) bit allocation

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Symbol | reserved | | | | DP2_SRP_EN_R | P2_A_SESS_VLD_EN_R | OTG_TMR_TIMEOUT_EN_R | B_SE0_SRP_EN_R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/S/C | R/S/C | R/S/C | R/S/C | R/S/C | R/S/C | R/S/C | R/S/C |
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| Symbol | B_SESS_END_EN_R | BDIS_ACON_EN_R | reserved | RMT_CONN_EN_R | ID_EN_R | DP_SRP_EN_R | A_B_SESS_VLD_EN_R | VBUS_VLD_EN_R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/S/C | R/S/C | R/S/C | R/S/C | R/S/C | R/S/C | R/S/C | R/S/C |

Table 21. OTG_INTR_EN_R - OTG Interrupt Enable Rise register (address set: F4h, clear: F6h) bit description

| Bit | Symbol | Description |
|---|---|---|
| 15 to 12 | - | reserved |
| 11 | DP2_SRP_EN_R | Enable interrupt when port 2 DP transitions from logic 0 to logic 1. |
| 10 | P2_A_SESS_VLD_EN_R | Enable interrupt when port 2 A_SESS_VLD transitions from logic 0 to logic 1 for the A-device. |
| 9 | OTG_TMR_TIMEOUT_EN_R | Enable interrupt on OTG timer time-out transition from logic 0 to logic 1. |
| 8 | B_SE0_SRP_EN_R | Enable interrupt when B_SE0_SRP transitions from logic 0 to logic 1. |
| 7 | B_SESS_END_EN_R | Enable interrupt when $V_{BUS}$ session end transitions from logic 0 to logic 1. |
| 6 | BDIS_ACON_EN_R | Enable interrupt when BDIS_ACON transitions from logic 0 to logic 1. |
| 5 | - | reserved |
| 4 | RMT_CONN_EN_R | Enable interrupt when RMT_CONN transitions from logic 0 to logic 1. |
| 3 | ID_EN_R | Enable interrupt when ID transitions from logic 0 to logic 1. |
| 2 | DP_SRP_EN_R | Enable interrupt when port 1 DP transitions from logic 0 to logic 1. |
| 1 | A_B_SESS_VLD_EN_R | Enable interrupt when AB-session valid transitions from logic 0 to logic 1. |
| 0 | VBUS_VLD_EN_R | Enable interrupt when VBUS_VLD transitions from logic 0 to logic 1. |

## 9.3 OTG Timer register

### 9.3.1 OTG Timer register

This is a 32-bit register organized as two 16-bit fields. These two fields have separate set and clear addresses. Table 22 shows the bit allocation of the register.

**Table 22. OTG_TMR - OTG Timer register (address low word set: F8h, low word clear: FAh; high word set: FCh, high word clear: FEh) bit allocation**

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Symbol | START_TMR | reserved | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/S/C | R/S/C | R/S/C | R/S/C | R/S/C | R/S/C | R/S/C | R/S/C |
| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| Symbol | TIMER_INIT_VALUE[23:16] | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/S/C | R/S/C | R/S/C | R/S/C | R/S/C | R/S/C | R/S/C | R/S/C |
| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| Symbol | TIMER_INIT_VALUE[15:8] | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/S/C | R/S/C | R/S/C | R/S/C | R/S/C | R/S/C | R/S/C | R/S/C |
| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Symbol | TIMER_INIT_VALUE[7:0] | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/S/C | R/S/C | R/S/C | R/S/C | R/S/C | R/S/C | R/S/C | R/S/C |

**Table 23. OTG_TMR - OTG Timer register (address low word set: F8h, low word clear: FAh; high word set: FCh, high word clear: FEh) bit description**

| Bit | Symbol | Description |
|---|---|---|
| 31 | START_TMR | This is the start/stop bit of the OTG timer. Writing logic 1 will cause the OTG timer to load TMR_INIT_VALUE into the counter and start to count. Writing logic 0 will stop the timer. This bit is automatically cleared when the OTG timer is timed out.<br>**0 —** Stop the timer.<br>**1 —** Start the timer. |
| 30 to 24 | - | reserved |
| 23 to 0 | TIMER_INIT_VALUE[23:0] | These bits define the initial value used by the OTG timer. The timer interval is 0.01 ms. Maximum time allowed is 167.772 s. |

# 10. Host controller-specific registers

Table 24 shows the bit description of the host controller-specific registers.

**Table 24. Overview of host controller-specific registers**

| Address | Register | Reset value | References |
|---|---|---|---|
| **EHCI operational registers** | | | |
| 8Ch | USBCMD | 0000 0000h | Section 10.1.1 on page 37 |
| 90h | USBSTS | 0000 0000h | Section 10.1.2 on page 37 |
| 94h | USBINTR | 0000 0000h | Section 10.1.3 on page 38 |
| 98h | FRINDEX | 0000 0000h | Section 10.1.4 on page 39 |
| 9Ch | CONFIGFLAG | 0000 0000h | Section 10.1.5 on page 40 |
| A0h | PORTSC1 | 0000 2000h | Section 10.1.6 on page 41 |
| A4h | ISO PTD Done Map | 0000h | Section 10.1.7 on page 42 |
| A6h | ISO PTD Skip Map | FFFFh | Section 10.1.8 on page 42 |
| A8h | ISO PTD Last PTD | 0000h | Section 10.1.9 on page 43 |
| AAh | INT PTD Done Map | 0000h | Section 10.1.10 on page 43 |
| ACh | INT PTD Skip Map | FFFFh | Section 10.1.11 on page 43 |
| AEh | INT PTD Last PTD | 0000h | Section 10.1.12 on page 43 |
| B0h | ATL PTD Done Map | 0000h | Section 10.1.13 on page 44 |
| B2h | ATL PTD Skip Map | FFFFh | Section 10.1.14 on page 44 |
| B4h | ATL PTD Last PTD | 0000h | Section 10.1.15 on page 44 |
| **Configuration registers** | | | |
| B6h | HW Mode Control | 0000h | Section 10.2.1 on page 45 |
| B8h | SW Reset | 0000h | Section 10.2.2 on page 46 |
| BAh | HcBufferStatus | 0000h | Section 10.2.3 on page 47 |
| BCh | HcDMAConfiguration | 0000h | Section 10.2.4 on page 48 |
| C0h | ATL Done Timeout | 0000 0000h | Section 10.2.5 on page 49 |
| C4h | Memory | 0000h | Section 10.2.6 on page 49 |
| C6h | Data Port | 0000h | Section 10.2.7 on page 50 |
| C8h | Edge Interrupt Count | 0000 000Fh | Section 10.2.8 on page 50 |
| 60h | DMA Data Port | - | Section 10.2.9 on page 51 |
| CCh | DMA Start Address | 0000h | Section 10.2.10 on page 51 |
| D0h | Power Down Control | 03E8 1BA0h | Section 10.2.11 on page 51 |
| **Interrupt registers** | | | |
| D4h | HcInterrupt | 0040h | Section 10.3.1 on page 53 |
| D6h | HcInterruptEnable | 0000h | Section 10.3.2 on page 55 |
| D8h | ISO IRQ MASK OR | 0000h | Section 10.3.3 on page 57 |
| DAh | INT IRQ MASK OR | 0000h | Section 10.3.4 on page 57 |
| DCh | ATL IRQ MASK OR | 0000h | Section 10.3.5 on page 57 |
| DEh | ISO IRQ MASK AND | 0000h | Section 10.3.6 on page 57 |
| E0h | INT IRQ MASK AND | 0000h | Section 10.3.7 on page 58 |
| E2h | ATL IRQ MASK AND | 0000h | Section 10.3.8 on page 58 |

## 10.1 EHCI operational registers

### 10.1.1 USBCMD register

The USB Command (USBCMD) register indicates the command to be executed by the serial host controller. Writing to this register causes a command to be executed.

Table 25 shows the USBCMD register bit allocation.

**Table 25. USBCMD - USB Command register (address 8Ch) bit allocation**

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Symbol | reserved[1] | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Bit | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
| Symbol | reserved[1] | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
| Symbol | reserved[1] | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Symbol | LHCR | reserved[1] | | | FLS[1:0] | | HCRESET | RS |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

[1] The reserved bits should always be written with the reset value.

**Table 26. USBCMD - USB Command register (address 8Ch) bit description**

| Bit | Symbol | Description[1] |
|---|---|---|
| 31 to 8 | - | reserved |
| 7 | LHCR | **Light host controller reset**: It allows the driver software to reset the EHCI controller without affecting the state of the ports or the relationship to the companion host controllers. To reset, write a one followed by a zero. |
| 6 to 4 | - | reserved |
| 3 to 2 | FLS[1:0] | **Frame list size**: Default 00b. This field specifies the size of the frame list. |
| 1 | HCRESET | **Host controller reset**: This control bit is used by the software to reset the host controller. The software writes one to this bit to reset. This bit is set to zero by the host controller when the reset process is complete. |
| 0 | RS | **Run/stop**: 1 = Run, 0 = Stop. When set, the host controller executes the schedule. |

[1] For details on register bit description, refer to *Enhanced Host Controller Interface Specification for Universal Serial Bus Rev. 1.0*.

### 10.1.2 USBSTS register

The USB Status (USBSTS) register indicates pending interrupts and various states of the host controller. The status resulting from a transaction on the serial bus is not indicated in this register. The software clears register bits by writing ones to them. The bit allocation is given in Table 27.

**Table 27. USBSTS - USB Status register (address 90h) bit allocation**

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Symbol | reserved[1] | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| **Bit** | **23** | **22** | **21** | **20** | **19** | **18** | **17** | **16** |
| Symbol | reserved[1] | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| **Bit** | **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** |
| Symbol | reserved[1] | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| Symbol | reserved[1] | | | | FLR | PCD | reserved[1] | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

[1] The reserved bits should always be written with the reset value.

**Table 28. USBSTS - USB Status register (address 90h) bit description**

| Bit | Symbol | Description[1] |
|---|---|---|
| 31 to 4 | - | reserved |
| 3 | FLR | **Frame list rollover**: The host controller sets this bit to logic 1 when the frame list index rolls over from its maximum value to logic 0. The exact value at which the rollover occurs depends on the FLS bits. |
| 2 | PCD | **Port change detect**: The host controller sets this bit to logic 1 when any port, where the PO bit is set to zero, has a bit transition from zero to one or the FPR bit transitions from zero to one as a result of a J-K transition detected on a suspended port. |
| 1 to 0 | - | reserved |

[1] For details on register bit description, refer to *Enhanced Host Controller Interface Specification for Universal Serial Bus Rev. 1.0*.

### 10.1.3 USBINTR register

The USB Interrupt (USBINTR) bit allocation is given in Table 29.

**Table 29. USBINTR - USB Interrupt register (address 94h) bit allocation**

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Symbol | reserved[1] | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| **Bit** | **23** | **22** | **21** | **20** | **19** | **18** | **17** | **16** |
| Symbol | reserved[1] | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Symbol | reserved[1] | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| Symbol | reserved[1] | | | | FLR_IE | PCD_IE | reserved[1] | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

[1] The reserved bits should always be written with the reset value.

**Table 30. USBINTR - USB Interrupt register (address 94h) bit description**

| Bit | Symbol | Description[1] |
|---|---|---|
| 31 to 4 | - | reserved |
| 3 | FLR_IE | **Frame list rollover interrupt enable**: When this bit is logic 1, and the Frame List Rollover bit in the USBSTS register is logic 1, the host controller will issue an interrupt. The interrupt is acknowledged by the software clearing the Frame List Rollover bit. |
| 2 | PCD_IE | **Port change detect interrupt enable**: When this bit is logic 1, and the Port Change Detect bit in the USBSTS register is logic 1, the host controller will issue an interrupt. The interrupt is acknowledged by the software clearing the Port Change Detect bit. |
| 1 to 0 | - | reserved |

[1] For details on register bit description, refer to *Enhanced Host Controller Interface Specification for Universal Serial Bus Rev. 1.0*.

### 10.1.4 FRINDEX register

The Frame Index (FRINDEX) register is used by the host controller to index into the periodic frame list. The register updates every 125 μs (once each microframe). Bits n to 3 are used to select a particular entry in the periodic frame list during periodic schedule execution. This register must be written as a double word. A byte or word write produces undefined results. This register must be written as a double word. A write to this register while the RS (Run/Stop) bit is set produces undefined results. Writes to this register also affect the SOF value.

The bit allocation is given in Table 31.

**Table 31. FRINDEX - Frame Index register (address: 98h) bit allocation**

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Symbol | reserved[1] | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| **Bit** | **23** | **22** | **21** | **20** | **19** | **18** | **17** | **16** |
| Symbol | reserved[1] | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Symbol | reserved[1] | | | | FRINDEX[13:8] | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| Symbol | FRINDEX[7:0] | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

[1] The reserved bits should always be written with the reset value.

**Table 32. FRINDEX - Frame Index register (address: 98h) bit description**

| Bit | Symbol | Description[1] |
|---|---|---|
| 31 to 14 | - | reserved |
| 13 to 0 | FRINDEX[13:0] | **Frame index**: Bits in this register are used for the frame number in the SOF packet and as the index into the frame list. The value in this register increments at the end of each time frame. |

[1] For details on register bit description, refer to *Enhanced Host Controller Interface Specification for Universal Serial Bus Rev. 1.0*.

### 10.1.5 CONFIGFLAG register

The bit allocation of the Configure Flag (CONFIGFLAG) register is given in Table 33.

**Table 33. CONFIGFLAG - Configure Flag register (address 9Ch) bit allocation**

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Symbol | reserved[1] | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| **Bit** | **23** | **22** | **21** | **20** | **19** | **18** | **17** | **16** |
| Symbol | reserved[1] | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| **Bit** | **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** |
| Symbol | reserved[1] | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| Symbol | reserved[1] | | | | | | | CF |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

[1] The reserved bits should always be written with the reset value.

**Table 34. CONFIGFLAG - Configure Flag register (address 9Ch) bit description**

| Bit | Symbol | Description[1] |
|---|---|---|
| 31 to 1 | - | reserved |
| 0 | CF | **Configure flag**: The host software sets this bit as the last action when it is configuring the host controller. This bit controls the default port-routing control logic. |

[1] For details on register bit description, refer to *Enhanced Host Controller Interface Specification for Universal Serial Bus Rev. 1.0*.

### 10.1.6 PORTSC1 register

The Port Status and Control (PORTSC) register (bit allocation: Table 35) is in the power well. It is reset by the hardware only when the auxiliary power is initially applied or in response to a host controller reset. The initial conditions of a port are:

- No peripheral connected
- Port disabled

If the port has power control, the software cannot change the state of the port until it sets port power bits. The software must not attempt to change the state of the port until the power is stable on the port (maximum delay is 20 ms from the transition).

**Table 35. PORTSC1 - Port Status and Control 1 register (address A0h) bit allocation**

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Symbol | reserved[1] | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| **Bit** | **23** | **22** | **21** | **20** | **19** | **18** | **17** | **16** |
| Symbol | reserved[1] | | | | PTC[3:0] | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| **Bit** | **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** |
| Symbol | reserved[1] | | PO | PP | LS[1:0] | | reserved[1] | PR |
| Reset | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| Symbol | SUSP | FPR | reserved[1] | | | PED | ECSC | ECCS |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

[1] The reserved bits should always be written with the reset value.

**Table 36. PORTSC1 - Port Status and Control 1 register (address A0h) bit description**

| Bit | Symbol | Description[1] |
|---|---|---|
| 31 to 20 | - | reserved |
| 19 to 16 | PTC[3:0] | **Port test control**: When this field is zero, the port is not operating in test mode. A non-zero value indicates that it is operating in test mode indicated by the value. |
| 15 to 14 | - | reserved |

**Table 36. PORTSC1 - Port Status and Control 1 register (address A0h) bit description** *...continued*

| Bit | Symbol | Description[1] |
|---|---|---|
| 13 | PO | **Port owner**: This bit unconditionally goes to logic 0 when the configured bit in the CONFIGFLAG register makes a logic 0 to logic 1 transition. This bit unconditionally goes to logic 1 whenever the configured bit is logic 0. |
| 12 | PP | **Port power**: This bit will enable the root hub for the upstream port. |
| 11 to 10 | LS[1:0] | **Line status**: This field reflects current logical levels of the DP (bit 11) and DM (bit 10) signal lines. |
| 9 | - | reserved |
| 8 | PR | **Port reset**: Write logic 1 to this bit to reset the bus. Write logic 0 to this bit to terminate the bus reset sequence.[2] |
| 7 | SUSP | **Suspend**: Logic 1 means the port is in the suspend state. Logic 0 means the port is not suspended.[2] |
| 6 | FPR | **Force port resume**: Logic 1 means resume detected or driven on the port. Logic 0 means no resume (K-state) detected or driven on the port.[2] |
| 5 to 3 | - | reserved |
| 2 | PED | **Port enabled/disabled**: Logic 1 means enable. Logic 0 means disable.[2] |
| 1 | ECSC | **Connect status change**: Logic 1 means change in ECCS. Logic 0 means no change.[2] |
| 0 | ECCS | **Current connect status**: Logic 1 indicates a peripheral is present on the port. Logic 0 indicates no device is present.[2] |

[1] For details on register bit description, refer to *Enhanced Host Controller Interface Specification for Universal Serial Bus Rev. 1.0*.

[2] These fields read logic 0, if the PP (Port Power) bit in register PORTSC1 is logic 0.

### 10.1.7 ISO PTD Done Map register

The bit description of the register is given in Table 37.

**Table 37. ISO PTD Done Map register (address A4h) bit description**
*Legend: * reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 15 to 0 | ISO_PTD_DONE_MAP[15:0] | R | 0000h* | **ISO PTD done map**: Done map for each of the 16 PTDs for the ISO transfer. |

This register represents a direct map of the done status of 16 PTDs. The bit corresponding to a particular PTD will be set to logic 1 as soon as that PTD execution is completed. Reading the done map register will clear all the bits that are set to logic 1, and the next reading will reflect the updated status of the new executed PTDs.

### 10.1.8 ISO PTD Skip Map register

Table 38 shows the bit description of the register.

**Table 38. ISO PTD Skip Map register (address A6h) bit description**
*Legend: * reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 15 to 0 | ISO_PTD_SKIP_MAP[15:0] | R/W | FFFFh* | **ISO PTD skip map**: Skip map for each of the 16 PTDs for the ISO transfer. |

When a bit in the PTD skip map is set to logic 1, that PTD will be skipped although its V bit may be set. The information in that PTD is not processed. When the PTD is in process, the software must not set this bit. After writing to this register, add 100 ns delay before reading it.

### 10.1.9 ISO PTD Last PTD register

Table 39 shows the bit description of the ISO PTD Last PTD register.

**Table 39. ISO PTD Last PTD register (address A8h) bit description**
*Legend: * reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 15 to 0 | ISO_PTD_LAST_PTD[15:0] | R/W | 0000h* | **ISO PTD last PTD**: Last PTD of the 16 PTDs.<br>**1h —** One PTD in ISO.<br>**2h —** Two PTDs in ISO.<br>**4h —** Three PTDs in ISO. |

Once the LastPTD bit corresponding to a PTD is set, this will be the last PTD processed (checking V = 1) in that PTD category. Subsequently, processing will restart with the first PTD of that group. This is useful to reduce the time in which all the PTDs, the respective memory space, would be checked, especially if only a few PTDs are defined.

### 10.1.10 INT PTD Done Map register

The bit description of the register is given in Table 40.

**Table 40. INT PTD Done Map register (address AAh) bit description**
*Legend: * reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 15 to 0 | INT_PTD_DONE_MAP[15:0] | R | 0000h* | **INT PTD done map**: Done map for each of the 16 PTDs for the INT transfer. |

This register represents a direct map of the done status of 16 PTDs. The bit corresponding to a particular PTD will be set to logic 1 as soon as that PTD execution is completed. Reading the done map register will clear all the bits that are set to logic 1, and the next reading will reflect the updated status of the new executed PTDs.

### 10.1.11 INT PTD Skip Map register

Table 41 shows the bit description of the INT PTD Skip Map register.

**Table 41. INT PTD Skip Map register (address ACh) bit description**
*Legend: * reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 15 to 0 | INT_PTD_SKIP_MAP[15:0] | R/W | FFFFh* | **INT PTD skip map**: Skip map for each of the 16 PTDs for the INT transfer. |

When a bit in the PTD skip map is set to logic 1, that PTD will be skipped although its V bit may be set. The information in that PTD is not processed. When the PTD is in process, the software must not set this bit. After writing to this register, add 100 ns delay before reading it.

### 10.1.12 INT PTD Last PTD register

The bit description of the register is given in Table 42.

**Table 42. INT PTD Last PTD register (address AEh) bit description**
*Legend: * reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 15 to 0 | INT_PTD_LAST_PTD[15:0] | R/W | 0000h* | **INT PTD last PTD**: Last PTD of the 16 PTDs.<br>**1h —** One PTD in INT.<br>**2h —** Two PTDs in INT.<br>**3h —** Three PTDs in INT. |

Once the LastPTD bit corresponding to a PTD is set, this will be the last PTD processed (checking V = 1) in that PTD category. Subsequently, processing will restart with the first PTD of that group. This is useful to reduce the time in which all the PTDs, the respective memory space, would be checked, especially if only a few PTDs are defined.

### 10.1.13 ATL PTD Done Map register

Table 43 shows the bit description of the ATL PTD Done Map register.

**Table 43. ATL PTD Done Map register (address B0h) bit description**
*Legend: * reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 15 to 0 | ATL_PTD_DONE_MAP[15:0] | R | 0000h* | **ATL PTD done map**: Done map for each of the 16 PTDs for the ATL transfer. |

This register represents a direct map of the done status of 16 PTDs. The bit corresponding to a particular PTD will be set to logic 1 as soon as that PTD execution is completed. Reading the done map register will clear all the bits that are set to logic 1, and the next reading will reflect the updated status of the new executed PTDs.

### 10.1.14 ATL PTD Skip Map register

The bit description of the register is given in Table 44.

**Table 44. ATL PTD Skip Map register (address B2h) bit description**
*Legend: * reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 15 to 0 | ATL_PTD_SKIP_MAP[15:0] | R/W | FFFFh* | **ATL PTD skip map**: Skip map for each of the 16 PTDs for the ATL transfer. |

When a bit in the PTD skip map is set to logic 1, that PTD will be skipped although its V bit may be set. The information in that PTD is not processed. When the PTD is in process, the software must not set this bit. After writing to this register, add 100 ns delay before reading it.

### 10.1.15 ATL PTD Last PTD register

The bit description of the ATL PTD Last PTD register is given in Table 45.

**Table 45. ATL PTD Last PTD register (address B4h) bit description**

*Legend: * reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 15 to 0 | ATL_PTD_LAST_PTD[15:0] | R/W | 0000h* | **ATL PTD last PTD**: Last PTD of the 16 PTDs.<br>**1h —** One PTD in ATL.<br>**2h —** Two PTDs in ATL.<br>**4h —** Three PTDs in ATL. |

Once the LastPTD bit corresponding to a PTD is set, this will be the last PTD processed (checking V = 1) in that PTD category. Subsequently, processing will restart with the first PTD of that group. This is useful to reduce the time in which all the PTDs, the respective memory space, would be checked, especially if only a few PTDs are defined.

## 10.2 Configuration registers

### 10.2.1 HW Mode Control register

Table 46 shows the bit allocation of the register.

**Remark:** Use single-byte write access when configuring registers in NAND or NOR 8-bit mode.

**Table 46. HW Mode Control - Hardware Mode Control register (address B6h) bit allocation**

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Symbol | reserved[1] | | | ID_PULLUP | DEV_DMA | COMN_INT | COMN_DMA | reserved[1] |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| Symbol | reserved[1] | DACK_POL | DREQ_POL | DATA_BUS_WIDTH | INTF_LOCK | INTR_POL | INTR_LEVEL | GLOBAL_INTR_EN |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

[1] The reserved bits should always be written with the reset value.

**Table 47. HW Mode Control - Hardware Mode Control register (address B6h) bit description**

| Bit | Symbol | Description |
|---|---|---|
| 15 to 13 | - | reserved |
| 12 | ID_PULLUP | **ID pull up**: Connects a pull-up to the ID line and enables sampling of the ID level. Disabling the ID line sampler will reduce the PHY power consumption.<br>**0 —** Disable sampling of the ID line.<br>**1 —** Enable sampling of the ID line. |
| 11 | DEV_DMA | **Device DMA**: When this bit and bit 9 are set, DC_DREQ and DC_DACK peripheral signals are routed to the DREQ and DACK pins. |
| 10 | COMN_INT | **Common INT**: When this bit is set, DC_INT will be routed to the INT pin. |
| 9 | COMN_DMA | **Common DMA**: When this bit and bit 11 are set, the DC_DREQ and DC_DACK peripheral signals are routed to the DREQ and DACK pins. |
| 8 to 7 | - | reserved |

**Table 47. HW Mode Control - Hardware Mode Control register (address B6h) bit description** *...continued*

| Bit | Symbol | Description |
|---|---|---|
| 6 | DACK_POL | **DACK polarity**:<br>**1 —** Indicates that the DACK input is active HIGH.<br>**0 —** Indicates that the DACK input is active LOW.<br>**Remark:** Value written to this bit must be the same as the value written to bit DACK_POL in the peripheral controller DMA Hardware register (address 3Ch). |
| 5 | DREQ_POL | **DREQ polarity**:<br>**1 —** Indicates that the DREQ output is active HIGH.<br>**0 —** Indicates that the DREQ output is active LOW.<br>**Remark:** Value written to this bit must be the same as the value written to bit DREQ_POL in the peripheral controller DMA Hardware register (address 3Ch). |
| 4 | DATA_BUS_WIDTH | **Data bus width**:<br>**0 —** Defines a 16-bit data bus width.<br>**1 —** Sets a 8-bit data bus width. |
| 3 | INTF_LOCK | **Interface lock**:<br>**0 —** Unlocks the bus interface.<br>**1 —** Locks the bus interface. |
| 2 | INTR_POL | **Interrupt polarity**:<br>**0 —** Active LOW<br>**1 —** Active HIGH<br>**Remark:** Value written to this bit must be the same as the value written to bit INTPOL in the peripheral controller Interrupt Configuration register (address 10h). |
| 1 | INTR_LEVEL | **Interrupt level**:<br>**0 —** INT is level triggered.<br>**1 —** INT is edge triggered. The pulse width depends on the NO_OF_CLK bits in the Edge Interrupt Count register.<br>**Remark:** Value written to this bit must be the same as the value written to bit INTLVL in the peripheral controller Interrupt Configuration register (address 10h). |
| 0 | GLOBAL_INTR_EN | **Global interrupt enable**: This bit must be set to logic 1 to enable INT signal assertion.<br>**0 —** INT assertion disabled. INT will never be asserted, regardless of other settings or INT events.<br>**1 —** INT assertion enabled. INT will be asserted according to the HcInterruptEnable register, and event setting and occurrence. |

### 10.2.2 SW Reset register

Table 48 shows the bit allocation of the register.

**Table 48. SW Reset - Software Reset register (address B8h) bit allocation**

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Symbol | reserved[1] | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Symbol | INTF_MODE[1:0] | | reserved[1] | | RESET_ATX | reserved[1] | RESET_HC | RESET_ALL |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R | R | R/W | R/W | R/W | R/W | R/W | R/W |

[1] The reserved bits should always be written with the reset value.

**Table 49. SW Reset - Software Reset register (address B8h) bit description**

| Bit | Symbol | Description |
|---|---|---|
| 15 to 8 | - | reserved |
| 7 to 6 | INTF_MODE[1:0] | **Interface mode**: When read:<br>**00b —** NAND mode<br>**01b —** Generic mode<br>**10b —** NOR mode<br>**11b —** SRAM mode<br>Writing to these bits have no effect. |
| 5 to 4 | - | reserved |
| 3 | RESET_ATX | **Reset ATX**: Reset both transceivers.<br>**0 —** No reset<br>**1 —** Enable reset<br>**Remark:** Writing logic 1 followed by logic 0 will enable the software reset to ATX. |
| 2 | - | reserved |
| 1 | RESET_HC | **Reset host controller**: Reset only host controller-specific registers.<br>**0 —** No reset<br>**1 —** Enable reset |
| 0 | RESET_ALL | **Reset all**: Reset all host controller and CPU interface registers.<br>**0 —** No reset<br>**1 —** Enable reset |

### 10.2.3 HcBufferStatus register

The HcBufferStatus register is used to indicate to the host controller that a particular PTD buffer (that is, ATL, INT, or ISO) contains at least one PTD that must be scheduled. Once the software sets the Buffer Filled bit of a particular transfer in the HcBufferStatus register, the host controller will start traversing through the PTD headers that are valid PTDs and not marked for skipping.

**Remark:** The software can set these bits during the initialization.

Table 50 shows the bit allocation of the HcBufferStatus register.

**Table 50. HcBufferStatus - Host Controller Buffer Status register (address BAh) bit allocation**

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Symbol | reserved[1] | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Symbol | reserved[1] | | | | | ISO_BUF_FILL | INT_BUF_FILL | ATL_BUF_FILL |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

[1] The reserved bits should always be written with the reset value.

**Table 51. HcBufferStatus - Host Controller Buffer Status register (address BAh) bit description**

| Bit | Symbol | Description |
|---|---|---|
| 15 to 3 | - | reserved |
| 2 | ISO_BUF_FILL | **ISO buffer filled**:<br>**1 —** Indicates one of the ISO PTDs is filled, and the ISO PTD area will be processed.<br>**0 —** Indicates there is no PTD in this area. Therefore, processing of ISO PTDs will be completely skipped. |
| 1 | INT_BUF_FILL | **INT buffer filled**:<br>**1 —** Indicates one of the INT PTDs is filled, and the INT PTD area will be processed.<br>**0 —** Indicates there is no PTD in this area. Therefore, processing of INT PTDs will be completely skipped. |
| 0 | ATL_BUF_FILL | **ATL buffer filled**:<br>**1 —** Indicates one of the ATL PTDs is filled, and the ATL PTD area will be processed.<br>**0 —** Indicates there is no PTD in this area. Therefore, processing of ATL PTDs will be completely skipped. |

### 10.2.4 HcDMAConfiguration register

The bit allocation of the HcDMAConfiguration register is given in Table 52.

**Table 52. HcDMAConfiguration - Host Controller Direct Memory Access Configuration register (address BCh) bit allocation**

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Symbol | DMA_COUNTER[23:16] | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| **Bit** | **23** | **22** | **21** | **20** | **19** | **18** | **17** | **16** |
| Symbol | DMA_COUNTER[15:8] | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| **Bit** | **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** |
| Symbol | DMA_COUNTER[7:0] | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Symbol | DIS_DMA_DATA_PORT | reserved[1] | | | BURST_LEN[1:0] | | ENABLE_DMA | DMA_READ_WRITE_SEL |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

[1] The reserved bits should always be written with the reset value.

**Table 53. HcDMAConfiguration - Host Controller Direct Memory Access Configuration register (address BCh) bit description**

| Bit | Symbol | Description |
|---|---|---|
| 31 to 8 | DMA_COUNTER [23:0] | **DMA counter**: The number of bytes to be transferred (read or write).<br>**Remark:** Different number of bursts will be generated for the same transfer length programmed in 8-bit and 16-bit modes because DMA_COUNTER is in number of bytes. |
| 7 | DIS_DMA_DATA_PORT | **Disable DMA data port**:<br>**0 —** Enable DMA data port.<br>**1 —** Disable DMA data port. |
| 6 to 4 | - | reserved |
| 3 to 2 | BURST_LEN[1:0] | **DMA burst length**:<br>**00 —** Single DMA burst<br>**01 —** 4-cycle DMA burst<br>**10 —** 8-cycle DMA burst<br>**11 —** 16-cycle DMA burst |
| 1 | ENABLE_DMA | **Enable DMA**:<br>**0 —** Terminate DMA<br>**1 —** Enable DMA |
| 0 | DMA_READ_WRITE_SEL | **DMA read or write select**: Indicates if the DMA operation is a write to or read from the ISP1763A.<br>**0 —** DMA write to the ISP1763A internal RAM.<br>**1 —** DMA read from the ISP1763A internal RAM. |

### 10.2.5 ATL Done Timeout register

The bit description of the ATL Done Timeout register is given in Table 54.

**Table 54. ATL Done Timeout register (address: C0h) bit description**
*Legend: * reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31 to 0 | ATL_DONE_TIMEOUT [31:0] | R/W | 0000 0000h* | **ATL done timeout**: This register determines the ATL done time-out interrupt. This register defines the time-out in seconds after which the ISP1763A asserts the INT line, if enabled. It is applicable to ATL done PTDs only. |

### 10.2.6 Memory register

The Memory register contains the base memory read or write address. This register must be set only before a first memory read cycle. Once written, the address will be latched and will be incremented for every read until a new address is written to change the address pointer.

The bit description of the register is given in Table 55.

**Table 55. Memory register (address C4h) bit allocation**

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Symbol | START_ADDR_MEM[15:8] | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| Symbol | START_ADDR_MEM[7:0] | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

**Table 56. Memory register (address C4h) bit description**

| Bit | Symbol | Description |
|---|---|---|
| 15 to 0 | START_ADDR_MEM[15:0] | **Start address for memory read or write cycles**: The start address for a series of memory read or write cycles at incremental addresses in a contiguous space. |

### 10.2.7 Data Port register

Table 57 shows the bit description of this register.

**Table 57. Data Port register (address: C6h) bit description**

*Legend: * reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 15 to 0 | DATA_PORT[15:0] | R/W | 0000h* | **Data port**: This register is used to access the host controller memory. |

### 10.2.8 Edge Interrupt Count register

Table 58 shows the bit allocation of the register.

**Table 58. Edge Interrupt Count register (address C8h) bit allocation**

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Symbol | MIN_WIDTH[7:0] | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| **Bit** | **23** | **22** | **21** | **20** | **19** | **18** | **17** | **16** |
| Symbol | reserved[1] | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| **Bit** | **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** |
| Symbol | NO_OF_CLK[15:8] | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| Symbol | NO_OF_CLK[7:0] | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

[1] The reserved bits should always be written with the reset value.

**Table 59. Edge Interrupt Count register (address C8h) bit description**

| Bit | Symbol | Description |
|---|---|---|
| 31 to 24 | MIN_WIDTH [7:0] | **Minimum width**: Indicates the minimum width between two edge interrupts in μSOFs (1 μSOF = 125 μs). This is not valid for level interrupts. A count of zero means that an interrupt occurs as and when an event occurs. |
| 23 to 16 | - | reserved |
| 15 to 0 | NO_OF_ CLK[15:0] | **Number of clocks**: Number of clocks that the edge interrupt must be kept asserted on the interface. 15 clocks of 30 MHz on POR if this register has a value of 000Fh. The default INT pulse width is approximately 500 ns. Zero is an invalid setting for these bits. |

### 10.2.9 DMA Data Port register

Table 60 for the bit description of the DMA Data Port register.

**Table 60. DMA Data Port register (address 60h) bit description**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 15 to 0 | DMA_DATA_PORT[15:0] | R/W | - | **DMA data port**: Access the memory through DMA. |

### 10.2.10 DMA Start Address register

This register defines the start address select for the DMA read and write operations. See Table 61 for bit description.

**Table 61. DMA Start Address register (address CCh) bit description**

*Legend: * reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 15 to 0 | START_ADDR_ DMA[15:0] | W | 0000h* | **Start address for DMA**: The start address for DMA read or write cycles. |

### 10.2.11 Power Down Control register

This register is used to turn off power to internal blocks of the ISP1763A to obtain maximum power savings. Table 62 shows the bit allocation of the register.

**Table 62. Power Down Control register (address D0h) bit allocation**

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Symbol | CLK_OFF_COUNTER[15:8] | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| **Bit** | **23** | **22** | **21** | **20** | **19** | **18** | **17** | **16** |
| Symbol | CLK_OFF_COUNTER[7:0] | | | | | | | |
| Reset | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| **Bit** | **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** |
| Symbol | P2_ FORCE_FS | P1_ FORCE_FS | reserved[1] | | PORT2_ PD | reserved[1] | P2_OC_EN | P1_OC_EN |
| Reset | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 1 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Symbol | ATX2_ PWRON | reserved[1] | REG_PWR | REG_ SUSP_ PWR | reserved[1] | P2_OTG_ EN | P1_OTG_ EN | HC_CLK_ EN |
| Reset | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

[1] The reserved bits should always be written with the reset value.

**Table 63. Power Down Control register (address D0h) bit description**

| Bit | Symbol | Description |
|---|---|---|
| 31 to 16 | CLK_OFF_ COUNTER[15:0] | **Clock off counter**: Determines the wake-up status duration after any wake-up event before the ISP1763A goes back into suspend mode. This time-out is applicable only if, during the given interval, the host controller is not programmed back to the normal functionality.<br>The setting of this register is based on the LazyClock frequency. |
| 15 | P2_FORCE_FS | **Port 2 force full-speed**:<br>**0 —** Port 2 is not forced into full-speed.<br>**1 —** Force port 2 into full-speed. |
| 14 | P1_FORCE_FS | **Port 1 force full-speed**:<br>**0 —** Port 1 is not forced into full-speed.<br>**1 —** Force port 1 into full-speed. |
| 13 to 12 | - | reserved |
| 11 | PORT2_PD | **Port 2 pull-down**: Controls port 2 pull-down resistors.<br>**0 —** Port 2 internal pull-down resistors are not connected.<br>**1 —** Port 2 internal pull-down resistors are connected. |
| 10 | - | reserved |
| 9 | P2_OC_EN | **Port 2 overcurrent enable**:<br>**0 —** Disable overcurrent on port 2.<br>**1 —** Enable overcurrent on port 2. |
| 8 | P1_OC_EN | **Port 1 overcurrent enable**:<br>**0 —** Disable overcurrent on port 1.<br>**1 —** Enable overcurrent on port 1. |
| 7 | ATX2_PWRON | **ATX2 power on**: Controls the ATX2 suspend.<br>**0 —** ATX2 is suspended. ATX2 macrocell is in suspend mode that draws minimal power from supplies.<br>**1 —** ATX2 is in normal operational mode. ATX2 macrocell draws normal current. |
| 6 | - | reserved |
| 5 | REG_PWR | **Regulator powered**: Controls the power-down mode.<br>**0 —** The device is set to power-down mode. Output of the regulator is shut down. The digital core has no power supply. The device can be woken up by a dummy read from the PIO interface.<br>**1 —** The regulator is in normal or suspend mode. |
| 4 | REG_SUSP_PWR | **Regulator suspend circuit powered**: Controls regulator suspend.<br>**0 —** The regulator is in normal or power-down mode.<br>**1 —** The regulator is in suspend mode. The digital core can draw limited current from the regulator. |
| 3 | - | reserved |

**Table 63. Power Down Control register (address D0h) bit description** *...continued*

| Bit | Symbol | Description |
|---|---|---|
| 2 | P2_OTG_EN | **Port 2 OTG comparators enable**: Controls the OTG detection for port 2.<br>**0 —** ATX OTG detection is powered off.<br>**1 —** ATX OTG detection is powered on.<br>Port 2 works with host mode only. This bit helps to detect DP asserted (DP2_SRP) and A-session valid for the A-device (A2_SESS_VLD). |
| 1 | P1_OTG_EN | **Port 1 OTG comparators enable**: Controls the OTG detection for port 1.<br>**0 —** ATX OTG detection is powered off. $V_{BUS}$ detection is still on for device mode.<br>**1 —** ATX OTG detection is powered on.<br>In device mode, to reduce power consumption, the OTG detection can be switched off. An independent circuit will assist port 1 to sense polarity changes on $V_{BUS}$. |
| 0 | HC_CLK_EN | **Host controller clock enabled**: Controls internal clocks during suspend.<br>**0 —** Clocks are disabled during suspend. This is the default value. Only the LazyClock of 100 kHz will be left running in suspend if this bit is logic 0. If clocks are stopped during suspend, CLKREADY INT will be generated when all clocks are running stable.<br>**1 —** All clocks are enabled even in suspend. |

## 10.3 Interrupt registers

### 10.3.1 HcInterrupt register

The bits of this register indicate the interrupt source, defining the events that determined the INT generation. Clearing the bits that were set because of the events listed is done by writing back logic 1 to the respective position. All bits must be reset before enabling new interrupt events. These bits will be set, regardless of the setting of bit GLOBAL_INTR_EN in the HW Mode Control register. Table 64 shows the bit allocation of the HcInterrupt register.

**Table 64. HcInterrupt - Host Controller Interrupt register (address D4h) bit description**

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Symbol | reserved[1] | | | | | OTG_IRQ | ISO_IRQ | ATL_IRQ |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| Symbol | INT_IRQ | CLK READY | HCSUSP | OPR_REG | DMAEOT INT | reserved[1] | SOFINT | MSOFINT |
| Reset | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

[1] The reserved bits should always be written with the reset value.

**Table 65. HcInterrupt - Host Controller Interrupt register (address D4h) bit description**

| Bit | Symbol | Description |
|---|---|---|
| 15 to 11 | - | reserved |
| 10 | OTG_IRQ | **OTG IRQ**: Indicates that an OTG event occurred. The INT line will be asserted if the respective enable bit in the HcInterruptEnable register is set.<br>**0 —** No OTG event.<br>**1 —** OTG event occurred.<br>For details, see Section 8.9.3. |
| 9 | ISO_IRQ | **ISO IRQ**: Indicates that an ISO PTD was completed, or the PTDs corresponding to the bits set in the ISO IRQ MASK AND or ISO IRQ MASK OR register bits combination were completed. The INT line will be asserted if the respective enable bit in the HcInterruptEnable register is set.<br>**0 —** No ISO PTD event occurred.<br>**1 —** ISO PTD event occurred.<br>For details, see Section 8.9.3. |
| 8 | ATL_IRQ | **ATL IRQ**: Indicates that an ATL PTD was completed, or the PTDs corresponding to the bits set in the ATL IRQ MASK AND or ATL IRQ MASK OR register bits combination were completed. The INT line will be asserted if the respective enable bit in the HcInterruptEnable register is set.<br>**0 —** No ATL PTD event occurred.<br>**1 —** ATL PTD event occurred.<br>For details, see Section 8.9.3. |
| 7 | INT_IRQ | **INT IRQ**: Indicates that an INT PTD was completed, or the PTDs corresponding to the bits set in the INT IRQ MASK AND or INT IRQ MASK OR register bits combination were completed. The INT line will be asserted if the respective enable bit in the HcInterruptEnable register is set.<br>**0 —** No INT PTD event occurred.<br>**1 —** INT PTD event occurred.<br>For details, see Section 8.9.3. |
| 6 | CLKREADY | **Clock ready**: Indicates that internal clock signals are running stable. The INT line will be asserted if the respective enable bit in the HcInterruptEnable register is set.<br>**0 —** No CLKREADY event has occurred.<br>**1 —** CLKREADY event occurred. |
| 5 | HCSUSP | **Host controller suspend**: Indicates that the host controller has entered suspend mode. The INT line will be asserted if the respective enable bit in the HcInterruptEnable register is set.<br>**0 —** The host controller did not enter suspend mode.<br>**1 —** The host controller entered suspend mode.<br>If the Interrupt Service Routine (ISR) accesses the ISP1763A, it will wake up for the time specified in bits 31 to 16 of the Power Down Control register. |

Table 65. HcInterrupt - Host Controller Interrupt register (address D4h) bit description *...continued*

| Bit | Symbol | Description |
|---|---|---|
| 4 | OPR_REG | **Operational registers**: Indicates an INT was generated because of at least one change in operational registers.<br>**0 —** No INT because of operational registers change.<br>**1 —** INT generated because of at least one change in operational registers. |
| 3 | DMAEOT INT | **DMA EOT interrupt**: Indicates the DMA transfer completion. The INT line will be asserted if the respective enable bit in the HcInterruptEnable register is set.<br>**0 —** No DMA transfer is completed.<br>**1 —** DMA transfer is completed. |
| 2 | - | reserved; value is zero just after reset and changes to one after a short while |
| 1 | SOFINT | **SOF interrupt**: The INT line will be asserted if the respective enable bit in the HcInterruptEnable register is set.<br>**0 —** No SOF event has occurred.<br>**1 —** SOF event has occurred. |
| 0 | MSOFINT | **μSOF ITL interrupt**: The INT line will be asserted if the respective enable bit in the HcInterruptEnable register is set.<br>**0 —** No μSOF event has occurred.<br>**1 —** μSOF event has occurred. |

### 10.3.2 HcInterruptEnable register

This register allows enabling or disabling of the INT generation because of various events as described in Table 66.

Table 66. HcInterruptEnable - Host Controller Interrupt Enable register (address D6h) bit allocation

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Symbol | reserved[1] | | | | | OTG_IRQ_E | ISO_IRQ_E | ATL_IRQ_E |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| Symbol | INT_IRQ_E | CLK READY_E | HCSUSP_E | OPR_REG_E | DMAEOT INT_E | reserved[1] | SOFINT_E | MSOFINT_E |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

[1] The reserved bits should always be written with the reset value.

**Table 67. HcInterruptEnable - Host Controller Interrupt Enable register (address D6h) bit description**

| Bit | Symbol | Description |
|---|---|---|
| 15 to 11 | - | reserved |
| 10 | OTG_IRQ_E | **OTG IRQ enable**: Controls the INT assertion because of events present in the OTG Interrupt Latch register.<br>**0 —** No INT will be asserted.<br>**1 —** INT will be asserted.<br>For details, see Section 8.9.3. |
| 9 | ISO_IRQ_E | **ISO IRQ enable**: Controls the INT assertion when one or more ISO PTDs matching the ISO IRQ Mask AND or ISO IRQ Mask OR register bits combination are completed.<br>**0 —** No INT will be asserted when ISO PTDs are completed.<br>**1 —** INT will be asserted.<br>For details, see Section 8.9.3. |
| 8 | ATL_IRQ_E | **ATL IRQ enable**: Controls the INT assertion when one or more ATL PTDs matching the ATL IRQ Mask AND or ATL IRQ Mask OR register bits combination are completed.<br>**0 —** No INT will be asserted when ATL PTDs are completed.<br>**1 —** INT will be asserted.<br>For details, see Section 8.9.3. |
| 7 | INT_IRQ_E | **INT IRQ enable**: Controls the INT assertion when one or more INT PTDs matching the INT IRQ Mask AND or INT IRQ Mask OR register bits combination are completed.<br>**0 —** No INT will be asserted when INT PTDs are completed.<br>**1 —** INT will be asserted.<br>For details, see Section 8.9.3. |
| 6 | CLKREADY_E | **Clock ready enable**: Enables the INT assertion when internal clock signals are running stable. Useful after wake-up.<br>**0 —** No INT will be generated after CLKREADY_E event.<br>**1 —** INT will be generated after a CLKREADY_E event. |
| 5 | HCSUSP_E | **Host controller suspend enable**: Enables the INT generation when the host controller enters suspend mode.<br>**0 —** No INT will be generated when the host controller enters suspend mode.<br>**1 —** INT will be generated when the host controller enters suspend mode. |
| 4 | OPR_REG_E | **Operational registers enable**: Controls the INT generation because of at least one change in operational registers.<br>**0 —** No INT will be generated on any change in operational registers.<br>**1 —** INT will be generated after a bit change, set by the software or hardware internally modified by the host controller, in the operational registers. |
| 3 | DMAEOT INT_E | **DMA EOT interrupt enable**: Controls assertion of INT on the DMA transfer completion.<br>**0 —** No INT will be generated when a DMA transfer is completed.<br>**1 —** INT will be asserted when a DMA transfer is completed. |
| 2 | - | reserved |
| 1 | SOFINT_E | **SOF interrupt enable**: Controls the INT generation at every SOF occurrence.<br>**0 —** No INT will be generated on SOF occurrence.<br>**1 —** INT will be asserted at every SOF. |
| 0 | MSOFINT_E | **μSOF interrupt enable**: Controls the INT generation at every μSOF occurrence.<br>**0 —** No INT will be generated on μSOF occurrence.<br>**1 —** INT will be asserted at every μSOF. |

### 10.3.3 ISO IRQ MASK OR register

Each bit of this register corresponds to one of the 16 ISO PTDs defined, and is a hardware INT mask for each PTD done map. See Table 68 for the bit description.

**Table 68. ISO IRQ MASK OR register (address D8h) bit description**
*Legend: * reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 15 to 0 | ISO_IRQ_ MASK_OR[15:0] | R/W | 0000h* | **ISO IRQ mask OR**: Represents a direct map for ISO PTDs 15 to 0.<br>**0 —** No OR condition defined between ISO PTDs<br>**1 —** The bits corresponding to certain PTDs are set to logic 1 to define a certain OR condition |

### 10.3.4 INT IRQ MASK OR register

Each bit of this register (see Table 69) corresponds to one of the 16 INT PTDs defined, and is a hardware INT mask for each PTD done map.

**Table 69. INT IRQ MASK OR register (address DAh) bit description**
*Legend: * reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 15 to 0 | INT_IRQ_ MASK_ OR[15:0] | R/W | 0000h* | **INT IRQ mask OR**: Represents a direct map for INT PTDs 15 to 0.<br>**0 —** No OR condition defined between INT PTDs 15 to 0.<br>**1 —** The bits corresponding to certain PTDs are set to logic 1 to define a certain OR condition. |

### 10.3.5 ATL IRQ MASK OR register

Each bit of this register corresponds to one of the 16 ATL PTDs defined, and is a hardware INT mask for each PTD done map. See Table 70 for the bit description.

**Table 70. ATL IRQ MASK OR register (address DCh) bit description**
*Legend: * reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 15 to 0 | ATL_IRQ_ MASK_OR [15:0] | R/W | 0000h* | **ATL IRQ mask OR**: Represents a direct map for ATL PTDs 15 to 0.<br>**0 —** No OR condition defined between ATL PTDs.<br>**1 —** The bits corresponding to certain PTDs are set to logic 1 to define a certain OR condition. |

### 10.3.6 ISO IRQ MASK AND register

Each bit of this register corresponds to one of the 16 ISO PTDs defined, and is a hardware INT mask for each PTD done map. See Table 71 for the bit description.

**Table 71. ISO IRQ MASK AND register (address DEh) bit description**
*Legend: * reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 15 to 0 | ISO_IRQ_ MASK_AND [15:0] | R/W | 0000h* | **ISO IRQ mask AND**: Represents a direct map for ISO PTDs 15 to 0.<br>**0 —** No AND condition defined between ISO PTDs.<br>**1 —** The bits corresponding to certain PTDs are set to logic 1 to define a certain AND condition between 16 INT PTDs. |

### 10.3.7 INT IRQ MASK AND register

Each bit of this register (see Table 72) corresponds to one of the 16 INT PTDs defined, and is a hardware INT mask for each PTD done map.

**Table 72. INT IRQ MASK AND register (address E0h) bit description**

*Legend: * reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 15 to 0 | INT_IRQ_MASK_AND [15:0] | R/W | 0000h* | **INT IRQ mask AND**: Represents a direct map for INT PTDs 15 to 0.<br>**0 —** No OR condition defined between INT PTDs.<br>**1 —** The bits corresponding to certain PTDs are set to logic 1 to define a certain AND condition between 16 INT PTDs. |

### 10.3.8 ATL IRQ MASK AND register

Each bit of this register corresponds to one of the 16 ATL PTDs defined, and is a hardware INT mask for each PTD done map. See Table 73 for the bit description.

**Table 73. ATL IRQ MASK AND register (address E2h) bit description**

*Legend: * reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 15 to 0 | ATL_IRQ_MASK_AND [15:0] | R/W | 0000h* | **INT IRQ mask AND**: Represents a direct map for ATL PTDs 15 to 0.<br>**0 —** No OR condition defined between ATL PTDs.<br>**1 —** The bits corresponding to certain PTDs are set to logic 1 to define a certain AND condition between 16 ATL PTDs. |

## 10.4 Proprietary Transfer Descriptor (PTD)

The standard EHCI data structures as described in *Enhanced Host Controller Interface Specification for Universal Serial Bus Rev. 1.0* are optimized for the bus master operation that is managed by the hardware state machine.

The PTD structures of the ISP1763A are translations of EHCI data structures that are optimized for the ISP1763A. They, however, still follow the basic EHCI architecture. This optimized form of EHCI data structures is necessary because the ISP1763A is a slave host controller and has no bus master capability.

EHCI manages schedules in two lists: periodic and asynchronous. Data structures are designed to provide the maximum flexibility required by USB, minimize memory traffic, and reduce hardware and software complexity. The ISP1763A controller executes transactions for devices by using a simple shared-memory schedule. This schedule consists of data structures organized into three lists.

**qISO —** Isochronous transfer

**qINTL —** Interrupt transfer

**qATL —** Asynchronous transfer; for the control and bulk transfers

The system software maintains two lists for the host controller: periodic and asynchronous.

The ISP1763A has a maximum of 16 ISO, 16 INTL, and 16 ATL PTDs. These PTDs are used as channels to transfer data from the shared memory to the USB bus. These channels are allocated and deallocated on receiving the transfer from the core USB driver.

Multiple transfers are scheduled to the shared memory for various endpoints by traversing the next link pointer provided by endpoint data structures, until it reaches the end of the endpoint list. There are three endpoint lists: one for ISO endpoints, and the other for INTL and ATL endpoints. If the schedule is enabled, the host controller executes the ISO schedule, followed by the INTL schedule, and then the ATL schedule.

These lists are traversed and scheduled by the software according to the EHCI traversal rule. The host controller executes scheduled ISO, INTL, and ATL PTDs. The completion of a transfer is indicated to the software by the interrupt that can be grouped under various PTDs by using the AND or OR registers that are available for each schedule type: ISO, INTL, and ATL. These registers are simple logic registers to decide the completion status of group and individual PTDs. When the logical conditions of the done bit are true in the shared memory, it means that PTD has completed.

There are four types of interrupts in the ISP1763A: ISO, INTL, ATL, and SOF. The latency can be programmed in multiples of μSOF (125 μs).

### 10.4.1 High-speed bulk IN and OUT

Table 74 shows the bit allocation of the high-speed bulk IN and OUT, asynchronous Transfer Descriptor.

**Table 74. High-speed bulk IN and OUT: bit allocation**

| Bit | 63 | 62 | 61 | 60 | 59 | 58 | 57 | 56 | 55 | 54 | 53 | 52 | 51 | 50 | 49 | 48 | 47 | 46 | 45 | 44 | 43 | 42 | 41 | 40 | 39 | 38 | 37 | 36 | 35 | 34 | 33 | 32 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DW7 | reserved | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| DW5 | reserved | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| DW3 | A | H | B | X | [1] | P | DT | Cerr [1:0] | | NakCnt[3:0] | | | | reserved | | | | NrBytesTransferred[14:0] (1 B to 20 kB for high-speed) | | | | | | | | | | | | | | |
| DW1 | reserved | | | | | | | | | | | | | | | | | S | EPType [1:0] | | Token [1:0] | | DeviceAddress[6:0] | | | | | | | EndPt[3:0] | | |

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DW6 | reserved | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| DW4 | reserved | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| DW2 | reserved | | | | RL[3:0] | | | [1] | DataStartAddress[15:0] | | | | | | | | | | | | | | reserved | | | | | | | | | |
| DW0 | [2] | Mult [1:0] | | MaxPacketLength[10:0] | | | | | | | | | | | NrBytesToTransfer[14:0] | | | | | | | | | | | | | | | [1] | | V |

[1] Reserved.

[2] EndPt[0].

**Table 75. High-speed bulk IN and OUT: bit description**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| **DW7** | | | | |
| 63 to 32 | reserved | - | - | - |
| **DW6** | | | | |
| 31 to 0 | reserved | - | - | - |
| **DW5** | | | | |
| 63 to 32 | reserved | - | - | - |
| **DW4** | | | | |
| 31 to 0 | reserved | - | 0 | Not applicable for asynchronous TD. |
| **DW3** | | | | |
| 63 | A | **SW —** sets<br>**HW —** resets | - | **Active**: Write the same value as that in V. |
| 62 | H | **HW —** writes | - | **Halt**: This bit corresponds to the Halt bit of the Status field of TD. |
| 61 | B | **HW —** writes | - | **Babble**: This bit corresponds to the Babble Detected bit in the Status field of iTD, siTD, or TD.<br>**1 —** When babbling is detected, A and V are set to 0. |
| 60 | X | **HW —** writes | - | **Error**: This bit corresponds to the Transaction Error bit in the Status field of iTD, siTD, or TD (Exec_Trans, the signal name is xacterr).<br>**0 —** No PID error.<br>**1 —** If there are PID errors, this bit is set to active. The A and V bits are also set to inactive. This transaction is retried three times. |
| | | **SW —** writes | - | **0 —** Before scheduling. |
| 59 | reserved | - | - | - |
| 58 | P | **SW —** writes<br>**HW —** updates | - | **Ping**: For high-speed transactions, this bit corresponds to the Ping state bit in the Status field of a TD.<br>**0 —** Ping is not set.<br>**1 —** Ping is set.<br>For the first time, the software sets the Ping bit to 0. For the successive asynchronous TD, the software sets the bit in asynchronous TD based on the state of the bit for the previous asynchronous TD of the same transfer, that is:<br>• The current asynchronous TD is completed with the Ping bit set.<br>• The next asynchronous TD will have its Ping bit set by the software. |
| 57 | DT | **HW —** updates<br>**SW —** writes | - | **Data toggle**: This bit is filled by the software to start a PTD. If NrBytesToTransfer[14:0] is not complete, the software needs to read this value and then write back the same value to continue. |
| 56 to 55 | Cerr[1:0] | **HW —** writes<br>**SW —** writes | - | **Error counter**: This field corresponds to the Cerr[1:0] field in TD. The default value of this field is zero for isochronous transactions.<br>**00 —** The transaction will not retry.<br>**11 —** The transaction will retry three times. The hardware will decrement these values. |

**Table 75. High-speed bulk IN and OUT: bit description** *…continued*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 54 to 51 | NakCnt[3:0] | **HW —** writes<br>**SW —** writes | - | **NAK counter**: This field corresponds to the NakCnt field in TD. The software writes for the initial PTD launch. The V bit is reset if NakCnt decrements to zero and RL is a non-zero value. It reloads from RL if transaction is ACK-ed. |
| 50 to 47 | reserved | - | - | - |
| 46 to 32 | NrBytes Transferred [14:0] | **HW —** writes<br>**SW —** writes 0000 | - | **Number of bytes transferred**: This field indicates the number of bytes sent or received for this transaction. If Mult[1:0] is greater than one, it is possible to store intermediate results in this field. |
| **DW2** | | | | |
| 31 to 29 | reserved | - | - | Set to 0 for asynchronous TD. |
| 28 to 25 | RL[3:0] | **SW —** writes | - | **Reload**: If RL[3:0] is set to 0h, the hardware ignores the NakCnt[3:0] value. RL[3:0] and NakCnt[3:0] are set to the same value before a transaction. |
| 24 | reserved | - | - | Always 0 for asynchronous TD. |
| 23 to 8 | DataStart Address[15:0] | **SW —** writes | - | **Data start address**: This is the start address for data that will be sent on or received from the USB bus. This is the internal memory address and not the direct CPU address.<br>RAM address = (CPU address − 400h) / 8 |
| 7 to 0 | reserved | - | - | - |
| **DW1** | | | | |
| 63 to 47 | reserved | - | - | Always 0 for asynchronous TD. |
| 46 | S | **SW —** writes | - | **Split**: This bit indicates whether a split transaction has to be executed:<br>**0 —** High-speed transaction<br>**1 —** Split transaction |
| 45 to 44 | EPType[1:0] | **SW —** writes | - | **Transaction type**:<br>**00 —** Control<br>**10 —** Bulk |
| 43 to 42 | Token[1:0] | **SW —** writes | - | **Token**: Identifies the token Packet Identifier (PID) for this transaction:<br>**00 —** OUT<br>**01 —** IN<br>**10 —** SETUP<br>**11 —** PING (written by the hardware only) |
| 41 to 35 | DeviceAddress [6:0] | **SW —** writes | - | **Device address**: This is the USB address of the function containing the endpoint that is referred to by this buffer. |
| 34 to 32 | EndPt[3:1] | **SW —** writes | - | **Endpoint**: This is the USB address of the endpoint within the function. |
| **DW0** | | | | |
| 31 | EndPt[0] | **SW —** writes | - | **Endpoint**: This is the USB address of the endpoint within the function. |
| 30 to 29 | Mult[1:0] | **SW —** writes | - | **Multiplier**: This field is a multiplier used by the host controller as the number of successive packets the host controller may submit to the endpoint in the current execution.<br>Set this field to 01b. You can also set it to 11b and 10b depending on your application. 00b is undefined. |

**Table 75. High-speed bulk IN and OUT: bit description** *…continued*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 28 to 18 | MaxPacket Length[10:0] | **SW —** writes | - | **Maximum packet length**: This field indicates the maximum number of bytes that can be sent to or received from an endpoint in a single data packet. The maximum packet size for a bulk transfer is 512 bytes. The maximum packet size for the isochronous transfer is also variable at any whole number. |
| 17 to 3 | NrBytesTo Transfer[14:0] | **SW —** writes | - | **Number of bytes to transfer**: This field indicates the number of bytes that can be transferred by this data structure. It is used to indicate the depth of the data field (1 B to 20 kB). |
| 2 to 1 | reserved | - | - | - |
| 0 | V | **SW —** sets<br>**HW —** resets | - | **Valid**:<br>**0 —** This bit is deactivated when the entire PTD is executed, or when a fatal error is encountered.<br>**1 —** The software updates to one when there is payload to be sent or received. The current PTD is active. |

### 10.4.2 High-speed isochronous IN and OUT

Table 76 shows the bit allocation of the high-speed isochronous IN and OUT, isochronous Transfer Descriptor (iTD).

**Table 76. High-speed isochronous IN and OUT: bit allocation**

| Bit | 63 | 62 | 61 | 60 | 59 | 58 | 57 | 56 | 55 | 54 | 53 | 52 | 51 | 50 | 49 | 48 | 47 | 46 | 45 | 44 | 43 | 42 | 41 | 40 | 39 | 38 | 37 | 36 | 35 | 34 | 33 | 32 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DW7 | ISOIN_7[11:0] | | | | | | | | | | | | ISOIN_6[11:0] | | | | | | | | | | | | ISOIN_5[11:4] | | | | | | | |
| DW5 | ISOIN_2[7:0] | | | | | | | | ISOIN_1[11:0] | | | | | | | | | | | | ISOIN_0[11:0] | | | | | | | | | | | |
| DW3 | A | H | B | reserved | | | | | | | | | | | | | | NrBytesTransferred[14:0] (1 B to 20 kB for high-speed) | | | | | | | | | | | | | | |
| DW1 | reserved | | | | | | | | | | | | | | | | | S | EP Type [1:0] | | Token [1:0] | | DeviceAddress[6:0] | | | | | | EndPt[3:0] | | |

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DW6 | ISOIN_5[3:0] | | | | ISOIN_4[11:0] | | | | | | | | | | | | ISOIN_3[11:0] | | | | | | | | | | | | ISOIN_2[11:8] | | | |
| DW4 | Status7[2:0] | | | Status6[2:0] | | | Status5[2:0] | | | Status4[2:0] | | | Status3[2:0] | | | Status2[2:0] | | | Status1[2:0] | | | Status0[2:0] | | | μSA[7:0] | | | | | | | |
| DW2 | reserved | | | | | | | | DataStartAddress[15:0] | | | | | | | | | | | | | | | | μFrame[7:0] | | | | | | | |
| DW0 | [2] | Mult [1:0] | | MaxPacketLength[10:0] | | | | | | | | | NrBytesToTransfer[14:0] | | | | | | | | | | | | | | | | | [1] | | V |

[1] Reserved.

[2] EndPt[0].

**Table 77. High-speed isochronous IN and OUT: bit description**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| **DW7** | | | | |
| 63 to 52 | ISOIN_7[11:0] | **HW —** writes | - | **Isochronous IN 7**: Bytes received during μSOF7, if μSA[7] is set to 1 and frame number is correct. |
| 51 to 40 | ISOIN_6[11:0] | **HW —** writes | - | **Isochronous IN 6**: Bytes received during μSOF6, if μSA[6] is set to 1 and frame number is correct. |
| 39 to 32 | ISOIN_5[11:4] | **HW —** writes | - | **Isochronous IN 5**: Bytes received during μSOF5 (bits 11 to 4), if μSA[5] is set to 1 and frame number is correct. |
| **DW6** | | | | |
| 31 to 28 | ISOIN_5[3:0] | **HW —** writes | - | **Isochronous IN 5**: Bytes received during μSOF5 (bits 3 to 0), if μSA[5] is set to 1 and frame number is correct. |
| 27 to 16 | ISOIN_4[11:0] | **HW —** writes | - | **Isochronous IN 4**: Bytes received during μSOF4, if μSA[4] is set to 1 and frame number is correct. |
| 15 to 4 | ISOIN_3[11:0] | **HW —** writes | - | **Isochronous IN 3**: Bytes received during μSOF3, if μSA[3] is set to 1 and frame number is correct. |
| 3 to 0 | ISOIN_2[11:8] | **HW —** writes | - | **Isochronous IN 2**: Bytes received during μSOF2 (bits 11 to 8), if μSA[2] is set to 1 and frame number is correct. |
| **DW5** | | | | |
| 63 to 56 | ISOIN_2[7:0] | **HW —** writes | - | **Isochronous IN 2**: Bytes received during μSOF2 (bits 7 to 0), if μSA[2] is set to 1 and frame number is correct. |
| 55 to 44 | ISOIN_1[11:0] | **HW —** writes | - | **Isochronous IN 1**: Bytes received during μSOF1, if μSA[1] is set to 1 and frame number is correct. |
| 43 to 32 | ISOIN_0[11:0] | **HW —** writes | - | **Isochronous IN 0**: Bytes received during μSOF0, if μSA[0] is set to 1 and frame number is correct. |
| **DW4** | | | | |
| 31 to 29 | Status7[2:0] | **HW —** writes | - | **Status 7**: ISO IN or OUT status at μSOF7 |
| 28 to 26 | Status6[2:0] | **HW —** writes | - | **Status 6**: ISO IN or OUT status at μSOF6 |
| 25 to 23 | Status5[2:0] | **HW —** writes | - | **Status 5**: ISO IN or OUT status at μSOF5 |
| 22 to 20 | Status4[2:0] | **HW —** writes | - | **Status 4**: ISO IN or OUT status at μSOF4 |
| 19 to 17 | Status3[2:0] | **HW —** writes | - | **Status 3**: ISO IN or OUT status at μSOF3 |
| 16 to 14 | Status2[2:0] | **HW —** writes | - | **Status 2**: ISO IN or OUT status at μSOF2 |
| 13 to 11 | Status1[2:0] | **HW —** writes | - | **Status 1**: ISO IN or OUT status at μSOF1 |
| 10 to 8 | Status0[2:0] | **HW —** writes | - | **Status 0**: Status of the payload on the USB bus for this μSOF after ISO has been delivered.<br>**Bit 0 —** Transaction error (IN and OUT)<br>**Bit 1 —** Babble (IN token only)<br>**Bit 2 —** Underrun (OUT token only) |
| 7 to 0 | μSA[7:0] | **SW —** writes (0 → 1)<br>**HW —** writes (1 → 0)<br>After processing | - | **μSOF active**: When the frame number of bits DW2[7:3] match the frame number of the USB bus, these bits are checked for 1 before they are sent for μSOF. For example: If μSA[7:0] = 1, 1, 1, 1, 1, 1, 1, 1: send ISO every μSOF of the entire millisecond. If μSA[7:0] = 0, 1, 0, 1, 0, 1, 0, 1: send ISO only on μSOF0, μSOF2, μSOF4, and μSOF6. |
| **DW3** | | | | |
| 63 | A | **SW —** sets | - | **Active**: This bit is the same as the Valid bit. |

**Table 77. High-speed isochronous IN and OUT: bit description** *...continued*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 62 | H | **HW —** writes | - | **Halt**: Only one bit for the entire millisecond. When this bit is set, the Valid bit is reset. The device decides to stall an endpoint. |
| 61 | B | **HW —** writes | - | **Babble**: Not applicable here. |
| 60 to 47 | reserved | - | 0 | Set to 0 for isochronous. |
| 46 to 32 | NrBytes Transferred [14:0] | **HW —** writes | - | **Number of bytes transferred**: This field indicates the number of bytes sent or received for this transaction. If Mult[1:0] is greater than one, it is possible to store intermediate results in this field. NrBytesTransferred[14:0] is 1 B to 20 kB per PTD. |
| **DW2** | | | | |
| 31 to 24 | reserved | - | 0 | Set to 0 for isochronous. |
| 23 to 8 | DataStart Address[15:0] | **SW —** writes | - | **Data start address**: This is the start address for data that will be sent on or received from the USB bus. This is the internal memory address and not the direct CPU address.<br>RAM address = (CPU address − 400h) / 8 |
| 7 to 0 | μFrame[7:0] | **SW —** writes | - | μ**Frame**:<br>**Bits 2 to 0 —** Don't care<br>**Bits 7 to 3 —** Frame number that this PTD will be sent for ISO OUT or IN |
| **DW1** | | | | |
| 63 to 47 | reserved | - | - | - |
| 46 | S | **SW —** writes | - | **Split**: This bit indicates whether a split transaction must be executed.<br>**0 —** High-speed transaction<br>**1 —** Split transaction |
| 45 to 44 | EPType[1:0] | **SW —** writes | - | **Endpoint type**:<br>**01 —** Isochronous |
| 43 to 42 | Token[1:0] | **SW —** writes | - | **Token**: This field indicates the token PID for this transaction:<br>**00 —** OUT<br>**01 —** IN |
| 41 to 35 | Device Address[6:0] | **SW —** writes | - | **Device address**: This is the USB address of the function containing the endpoint that is referred to by this buffer. |
| 34 to 32 | EndPt[3:1] | **SW —** writes | - | **Endpoint**: This is the USB address of the endpoint within the function. |
| **DW0** | | | | |
| 31 | EndPt[0] | **SW —** writes | - | **Endpoint**: This is the USB address of the endpoint within the function. |
| 30 to 29 | Mult[1:0] | **SW —** writes | - | **Multiplier**: This field is a multiplier counter used by the host controller as the number of successive packets the host controller may submit to the endpoint in the current execution.<br>For details, refer to Appendix D of *Enhanced Host Controller Interface Specification for Universal Serial Bus Rev. 1.0*. |
| 28 to 18 | MaxPacket Length[10:0] | **SW —** writes | - | **Maximum packet length**: This field indicates the maximum number of bytes that can be sent to or received from the endpoint in a single data packet. The maximum packet size for an isochronous transfer is 1024 bytes. The maximum packet size for the isochronous transfer is also variable at any whole number. |

**Table 77. High-speed isochronous IN and OUT: bit description** *…continued*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 17 to 3 | NrBytesTo Transfer[14:0] | **SW —** writes | - | **Number of bytes to transfer**: This field indicates the number of bytes that can be transferred by this data structure. It is used to indicate the depth of the data field (1 B to 20 kB). |
| 2 to 1 | reserved | - | - | - |
| 0 | V | **HW —** resets<br>**SW —** sets | - | **Valid**:<br>**0 —** This bit is deactivated when the entire PTD is executed, or when a fatal error is encountered.<br>**1 —** The software updates to one when there is payload to be sent or received. The current PTD is active. |

### 10.4.3 High-speed interrupt IN and OUT

Table 78 shows the bit allocation of the high-speed interrupt IN and OUT, periodic Transfer Descriptor (pTD).

**Table 78. High-speed interrupt IN and OUT: bit allocation**

| Bit | 63 | 62 | 61 | 60 | 59 | 58 | 57 | 56 | 55 | 54 | 53 | 52 | 51 | 50 | 49 | 48 | 47 | 46 | 45 | 44 | 43 | 42 | 41 | 40 | 39 | 38 | 37 | 36 | 35 | 34 | 33 | 32 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DW7 | INT_IN_7[11:0] | | | | | | | | | | | | INT_IN_6[11:0] | | | | | | | | | | | | INT_IN_5[11:4] | | | | | | | |
| DW5 | INT_IN_2[7:0] | | | | | | | | INT_IN_1[11:0] | | | | | | | | | | | | INT_IN_0[11:0] | | | | | | | | | | | |
| DW3 | A | H | reserved | | | | DT | Cerr [1:0] | reserved | | | | | | | | | NrBytesTransferred[14:0] (1 B to 20 kB for high-speed) | | | | | | | | | | | | | | | |
| DW1 | reserved | | | | | | | | | | | | | | | | | S | EP Type [1:0] | | Token [1:0] | | DeviceAddress[6:0] | | | | | | | EndPt[3:0] | | | |

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DW6 | INT_IN_5[3:0] | | | | INT_IN_4[11:0] | | | | | | | | | | | | INT_IN_3[11:0] | | | | | | | | | | | | INT_IN_2[11:8] | | | |
| DW4 | Status7[2:0] | | | Status6[2:0] | | | Status5[2:0] | | | Status4[2:0] | | | Status3[2:0] | | | Status2[2:0] | | | Status1[2:0] | | | Status0[2:0] | | | | | | | μSA[7:0] | | | |
| DW2 | reserved | | | | | | | | DataStartAddress[15:0] | | | | | | | | | | | | | | | | | | | | | μFrame[7:0] | | | |
| DW0 | [2] | Mult [1:0] | | | MaxPacketLength[10:0] | | | | | | | | | | | | NrBytesToTransfer[14:0] | | | | | | | | | | | | | [1] | | V |

[1] Reserved.

[2] EndPt[0].

**Table 79. High-speed interrupt IN and OUT: bit description**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| **DW7** | | | | |
| 63 to 52 | INT_IN_7[11:0] | **HW —** writes | - | **Interrupt IN 7**: Bytes received during μSOF7, if μSA[7] is set to 1 and frame number is correct. |
| 51 to 40 | INT_IN_6[11:0] | **HW —** writes | - | **Interrupt IN 6**: Bytes received during μSOF6, if μSA[6] is set to 1 and frame number is correct. |
| 39 to 32 | INT_IN_5[11:4] | **HW —** writes | - | **Interrupt IN 5**: Bytes received during μSOF5 (bits 11 to 4), if μSA[5] is set to 1 and frame number is correct. |
| **DW6** | | | | |
| 31 to 28 | INT_IN_5[3:0] | **HW —** writes | - | **Interrupt IN 5**: Bytes received during μSOF5 (bits 3 to 0), if μSA[5] is set to 1 and frame number is correct. |
| 27 to 16 | INT_IN_4[11:0] | **HW —** writes | - | **Interrupt IN 4**: Bytes received during μSOF4, if μSA[4] is set to 1 and frame number is correct. |
| 15 to 4 | INT_IN_3[11:0] | **HW —** writes | - | **Interrupt IN 3**: Bytes received during μSOF3, if μSA[3] is set to 1 and frame number is correct. |
| 3 to 0 | INT_IN_2[11:8] | **HW —** writes | - | **Interrupt IN 2**: Bytes received during μSOF2 (bits 11 to 8), if μSA[2] is set to 1 and frame number is correct. |
| **DW5** | | | | |
| 63 to 56 | INT_IN_2[7:0] | **HW —** writes | - | **Interrupt IN 2**: Bytes received during μSOF2 (bits 7 to 0), if μSA[2] is set to 1 and frame number is correct. |
| 55 to 44 | INT_IN_1[11:0] | **HW —** writes | - | **Interrupt IN 1**: Bytes received during μSOF1, if μSA[1] is set to 1 and frame number is correct. |
| 43 to 32 | INT_IN_0[11:0] | **HW —** writes | - | **Interrupt IN 0**: Bytes received during μSOF0, if μSA[0] is set to 1 and frame number is correct. |
| **DW4** | | | | |
| 31 to 29 | Status7[2:0] | **HW —** writes | - | **Status 7**: INT IN or OUT status of μSOF7 |
| 28 to 26 | Status6[2:0] | **HW —** writes | - | **Status 6**: INT IN or OUT status of μSOF6 |
| 25 to 23 | Status5[2:0] | **HW —** writes | - | **Status 5**: INT IN or OUT status of μSOF5 |
| 22 to 20 | Status4[2:0] | **HW —** writes | - | **Status 4**: INT IN or OUT status of μSOF4 |
| 19 to 17 | Status3[2:0] | **HW —** writes | - | **Status 3**: INT IN or OUT status of μSOF3 |
| 16 to 14 | Status2[2:0] | **HW —** writes | - | **Status 2**: INT IN or OUT status of μSOF2 |
| 13 to 11 | Status1[2:0] | **HW —** writes | - | **Status 1**: INT IN or OUT status of μSOF1 |
| 10 to 8 | Status0[2:0] | **HW —** writes | - | **Status 0**: Status of the payload on the USB bus for this μSOF after INT has been delivered.<br>**Bit 0 —** Transaction error (IN and OUT)<br>**Bit 1 —** Babble (IN token only)<br>**Bit 2 —** Underrun (OUT token only) |
| 7 to 0 | μSA[7:0] | **SW —** writes (0 → 1)<br>**HW —** writes (1 → 0)<br>After processing | - | **μSA[7:0]**: When the frame number of bits DW2[7:3] match the frame number of the USB bus, these bits are checked for 1 before they are sent for μSOF. For example: When μSA[7:0] = 1, 1, 1, 1, 1, 1, 1, 1: send INT for every μSOF of the entire millisecond. When μSA[7:0] = 0, 1, 0, 1, 0, 1, 0, 1: send INT for μSOF0, μSOF2, μSOF4, and μSOF6. When μSA[7:0] = 1, 0, 0, 0, 1, 0, 0, 0: send INT for every fourth μSOF. |

**Table 79. High-speed interrupt IN and OUT: bit description** *...continued*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| **DW3** | | | | |
| 63 | A | **HW —** writes<br>**SW —** writes | - | **Active**: Write the same value as that in V. |
| 62 | H | **HW —** writes | - | **Halt**: Transaction is halted. |
| 61 to 58 | reserved | - | - | - |
| 57 | DT | **HW —** writes<br>**SW —** writes | - | **Data toggle**: Set the Data Toggle bit to start the PTD. The software writes the current transaction toggle value. The hardware writes the next transaction toggle value. |
| 56 to 55 | Cerr[1:0] | **HW —** writes<br>**SW —** writes | - | **Error counter**: This field corresponds to the Cerr[1:0] field in TD. The default value of this field is zero for isochronous transactions. |
| 54 to 47 | reserved | - | - | - |
| 46 to 32 | NrBytes Transferred [14:0] | **HW —** writes | - | **Number of bytes transferred**: This field indicates the number of bytes sent or received for this transaction. If Mult[1:0] is greater than one, it is possible to store intermediate results in this field. |
| **DW2** | | | | |
| 31 to 24 | reserved | - | - | - |
| 23 to 8 | DataStart Address[15:0] | **SW —** writes | - | **Data start address**: This is the start address for data that will be sent on or received from the USB bus. This is the internal memory address and not the direct CPU address.<br>RAM address = (CPU address − 400h) / 8 |
| 7 to 0 | μFrame[7:0] | **SW —** writes | - | **μFrame**: Bits 7 to 3 represent the polling rate in milliseconds.<br>The INT polling rate is defined as $2^{(b-1)}$ μSOF, where b is 1 to 9.<br>When b is 1, 2, 3, or 4, use μSA to define polling because the rate is equal to or less than 1 ms. Bits 7 to 3 are set to 0. Polling checks μSA bits for μSOF rates. See Table 80. |
| **DW1** | | | | |
| 63 to 47 | reserved | - | - | - |
| 46 | S | **SW —** writes | - | **Split**: This bit indicates if a split transaction has to be executed:<br>**0 —** High-speed transaction<br>**1 —** Split transaction |
| 45 to 44 | EPType[1:0] | **SW —** writes | - | **Endpoint type**:<br>**11 —** Interrupt |
| 43 to 42 | Token[1:0] | **SW —** writes | - | **Token**: This field indicates the token PID for this transaction:<br>**00 —** OUT<br>**01 —** IN |
| 41 to 35 | DeviceAddress [6:0] | **SW —** writes | - | **Device address**: This is the USB address of the function containing the endpoint that is referred to by the buffer. |
| 34 to 32 | EndPt[3:1] | **SW —** writes | - | **Endpoint**: This is the USB address of the endpoint within the function. |
| **DW0** | | | | |
| 31 | EndPt[0] | **SW —** writes | - | **Endpoint**: This is the USB address of the endpoint within the function. |

**Table 79. High-speed interrupt IN and OUT: bit description** *...continued*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 30 to 29 | Mult[1:0] | **SW —** writes | - | **Multiplier**: This field is a multiplier counter used by the host controller as the number of successive packets the host controller may submit to the endpoint in the current execution.<br>Set this field to 01b. You can also set it to 11b and 10b depending on your application. 00b is undefined. |
| 28 to 18 | MaxPacket Length[10:0] | **SW —** writes | - | **Maximum packet length**: This field indicates the maximum number of bytes that can be sent to or received from the endpoint in a single data packet. |
| 17 to 3 | NrBytesTo Transfer[14:0] | **SW —** writes | - | **Number of bytes to transfer**: This field indicates the number of bytes that can be transferred by this data structure. It is used to indicate the depth of the data field (1 B to 20 kB). |
| 2 to 1 | reserved | - | - | - |
| 0 | V | **SW —** sets<br>**HW —** resets | - | **Valid**:<br>**0 —** This bit is deactivated when the entire PTD is executed, or when a fatal error is encountered.<br>**1 —** The software updates to one when there is payload to be sent or received. The current PTD is active. |

**Table 80. Microframe description**

| b | Rate | μFrame[7:3] | μSA[7:0] |
|---|---|---|---|
| 1 | 1 μSOF | 0 0000 | 1111 1111 |
| 2 | 2 μSOF | 0 0000 | 1010 1010 or 0101 0101 |
| 3 | 4 μSOF | 0 0000 | any 2 bits set |
| 4 | 1 ms | 0 0000 | any 1 bit set |
| 5 | 2 ms | 0 0001 | any 1 bit set |
| 6 | 4 ms | 0 0010 to 0 0011 | any 1 bit set |
| 7 | 8 ms | 0 0100 to 0 0111 | any 1 bit set |
| 8 | 16 ms | 0 1000 to 0 1111 | any 1 bit set |
| 9 | 32 ms | 1 0000 to 1 1111 | any 1 bit set |

### 10.4.4 Start and complete split for bulk

Table 81 shows the bit allocation of Start Split (SS) and Complete Split (CS) for bulk, asynchronous Start Split, and Complete Split (SS/CS) Transfer Descriptor.

**Table 81. Start and complete split for bulk: bit allocation**

| Bit | 63 | 62 | 61 | 60 | 59 | 58 | 57 | 56 | 55 | 54 | 53 | 52 | 51 | 50 | 49 | 48 | 47 | 46 | 45 | 44 | 43 | 42 | 41 | 40 | 39 | 38 | 37 | 36 | 35 | 34 | 33 | 32 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DW7 | reserved | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| DW5 | reserved | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| DW3 | A | H | B | X | SC | [1] | DT | Cerr [1:0] | | NakCnt[3:0] | | | | reserved | | | | NrBytesTransferred[14:0] | | | | | | | | | | | | | | |
| DW1 | HubAddress[6:0] | | | | | | | PortNumber[6:0] | | | | | | | SE[1:0] | | Re tire | S | EP Type [1:0] | | Token [1:0] | | DeviceAddress[6:0] | | | | | | | EndPt[3:0] | | |
| **Bit** | **31** | **30** | **29** | **28** | **27** | **26** | **25** | **24** | **23** | **22** | **21** | **20** | **19** | **18** | **17** | **16** | **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| DW6 | reserved | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| DW4 | reserved | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| DW2 | reserved | | | RL[3:0] | | | | [1] | DataStartAddress[15:0] | | | | | | | | | | | | | | reserved | | | | | | | | | |
| DW0 | [2] | [1] | | MaxPacketLength[10:0] | | | | | | | | | | | NrBytesToTransfer[14:0] | | | | | | | | | | | | | | | [1] | | V |

[1] Reserved.

[2] EndPt[0].

**Table 82. Start and complete split for bulk: bit description**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| **DW7** | | | | |
| 63 to 32 | reserved | - | - | - |
| **DW6** | | | | |
| 31 to 0 | reserved | - | - | - |
| **DW5** | | | | |
| 63 to 32 | reserved | - | - | - |
| **DW4** | | | | |
| 31 to 0 | reserved | - | - | - |
| **DW3** | | | | |
| 63 | A | **SW —** sets<br>**HW —** resets | - | **Active**: Write the same value as that in V. |
| 62 | H | **HW —** writes | - | **Halt**: This bit corresponds to the Halt bit of the Status field of TD. |
| 61 | B | **HW —** writes | - | **Babble**: This bit corresponds to the Babble Detected bit in the Status field of iTD, siTD, or TD.<br>**1 —** When babbling is detected, A and V are set to 0. |
| 60 | X | **HW —** writes | - | **Transaction error**: This bit corresponds to the Transaction Error bit in the Status field. |
| | | **SW —** writes | - | **0 —** Before scheduling |
| 59 | SC | **SW —** writes 0<br>**HW —** updates | - | **Start/complete**:<br>**0 —** Start split<br>**1 —** Complete split |
| 58 | reserved | - | - | - |
| 57 | DT | **HW —** writes<br>**SW —** writes | - | **Data toggle**: Set the Data Toggle bit to start for the PTD. |
| 56 to 55 | Cerr[1:0] | **HW —** updates<br>**SW —** writes | - | **Error counter**: This field contains the error count for asynchronous start and complete split (SS/CS) TD. When an error has no response or bad response, Cerr[1:0] will be decremented to zero and then Valid will be set to zero. A NAK or NYET will reset Cerr[1:0]. For details, refer to Section 4.12.1.2 of *Enhanced Host Controller Interface Specification for Universal Serial Bus Rev. 1.0*.<br>If retry has insufficient time at the beginning of a new SOF, the first PTD must be this retry. This can be accomplished if aperiodic PTD is not advanced. |
| 54 to 51 | NakCnt[3:0] | **HW —** writes<br>**SW —** writes | - | **NAK counter**: The V bit is reset if NakCnt decrements to zero and RL is a non-zero value. Not applicable to isochronous split transactions. |
| 50 to 47 | reserved | - | - | - |
| 46 to 32 | NrBytes Transferred[14:0] | **HW —** writes | - | **Number of bytes transferred**: This field indicates the number of bytes sent or received for this transaction. |
| **DW2** | | | | |
| 31 to 29 | reserved | - | - | - |

**Table 82. Start and complete split for bulk: bit description** *…continued*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 28 to 25 | RL[3:0] | **SW —** writes | - | **Reload**: If RL is set to 0h, the hardware ignores the NakCnt value. Set RL and NakCnt to the same value before a transaction. For full-speed and low-speed transactions, set this field to 0000b. Not applicable to isochronous start split and complete split. |
| 23 to 8 | DataStart Address[15:0] | **SW —** writes | - | **Data start address**: This is the start address for data that will be sent on or received from the USB bus. This is the internal memory address and not the direct CPU address.<br>RAM address = (CPU address − 400h) / 8 |
| 7 to 0 | reserved | - | - | - |
| **DW1** | | | | |
| 63 to 57 | HubAddress[6:0] | **SW —** writes | - | **Hub address**: This indicates the hub address. |
| 56 to 50 | PortNumber[6:0] | **SW —** writes | - | **Port number**: This indicates the port number of the hub or embedded TT. |
| 49 to 48 | SE[1:0] | **SW —** writes | - | This depends on the endpoint type and direction. It is valid only for split transactions. Table 83 applies to start split and complete split only. |
| 47 | Retire | **SW —** writes | - | **Retire**: Set to 1 to abort this PTD. |
| 46 | S | **SW —** writes | - | **Split**: This bit indicates whether a split transaction has to be executed:<br>**0 —** High-speed transaction<br>**1 —** Split transaction |
| 45 to 44 | EPType[1:0] | **SW —** writes | - | **Endpoint type**:<br>**00 —** Control<br>**10 —** Bulk |
| 43 to 42 | Token[1:0] | **SW —** writes | - | **Token**: This field indicates the PID for this transaction.<br>**00 —** OUT<br>**01 —** IN<br>**10 —** SETUP |
| 41 to 35 | DeviceAddress[6:0] | **SW —** writes | - | **Device address**: This is the USB address of the function containing the endpoint that is referred to by this buffer. |
| 34 to 32 | EndPt[3:1] | **SW —** writes | - | **Endpoint**: This is the USB address of the endpoint within the function. |
| **DW0** | | | | |
| 31 | EndPt[0] | **SW —** writes | - | **Endpoint**: This is the USB address of the endpoint within the function. |
| 30 to 29 | reserved | - | - | - |
| 28 to 18 | MaximumPacket Length[10:0] | **SW —** writes | - | **Maximum packet length**: This field indicates the maximum number of bytes that can be sent to or received from an endpoint in a single data packet. The maximum packet size for full-speed is 64 bytes as defined in *Universal Serial Bus Specification Rev. 2.0*. |

**Table 82. Start and complete split for bulk: bit description** *...continued*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 17 to 3 | NrBytesTo Transfer[14:0] | **SW —** writes | - | **Number of bytes to transfer**: This field indicates the number of bytes that can be transferred by this data structure. It is used to indicate the depth of the data field. |
| 2 to 1 | reserved | - | - | - |
| 0 | V | **SW —** sets<br>**HW —** resets | - | **Valid**:<br>**0 —** This bit is deactivated when the entire PTD is executed, or when a fatal error is encountered.<br>**1 —** The software updates to one when there is payload to be sent or received. The current PTD is active. |

**Table 83. SE description**

| Bulk | Control | SE1 | SE[0] | Remark |
|---|---|---|---|---|
| I/O | I/O | 1 | 0 | low-speed |
| I/O | I/O | 0 | 0 | full-speed |

### 10.4.5 Start and complete split for isochronous

Table 84 shows the bit allocation for start and complete split for isochronous, split isochronous Transfer Descriptor (siTD).

**Table 84. Start and complete split for isochronous: bit allocation**

| Bit | 63 | 62 | 61 | 60 | 59 | 58 | 57 | 56 | 55 | 54 | 53 | 52 | 51 | 50 | 49 | 48 | 47 | 46 | 45 | 44 | 43 | 42 | 41 | 40 | 39 | 38 | 37 | 36 | 35 | 34 | 33 | 32 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DW7 | reserved | | | | | | | | | | | | | | | | | | | | | | | | ISO_IN_7[7:0] | | | | | | | |
| DW5 | ISO_IN_2[7:0] | | | | | | | | ISO_IN_1[7:0] | | | | | | | | ISO_IN_0[7:0] | | | | | | | | μSCS[7:0] | | | | | | | |
| DW3 | A | H | B | X | SC | [1] | DT | reserved | | | | | | | | | | | | | NrBytesTransferred[11:0] | | | | | | | | | | | |
| DW1 | HubAddress[6:0] | | | | | | | PortNumber[6:0] | | | | | | | reserved | | | S | EP Type [1:0] | | Token [1:0] | | DeviceAddress[6:0] | | | | | | | EndPt[3:0] | | | |

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DW6 | ISO_IN_6[7:0] | | | | | | | | ISO_IN_5[7:0] | | | | | | | | ISO_IN_4[7:0] | | | | | | | | ISO_IN_3[7:0] | | | | | | | |
| DW4 | Status7[2:0] | | | Status6[2:0] | | | Status5[2:0] | | | Status4[2:0] | | | Status3[2:0] | | | Status2[2:0] | | | Status1[2:0] | | | Status0[2:0] | | | μSA[7:0] | | | | | | | |
| DW2 | reserved | | | | | | | | DataStartAddress[15:0] | | | | | | | | | | | | | | | | μFrame[7:0] (full-speed) | | | | | | | |
| DW0 | [2] | | [1] | | | TT_MPS_Len[10:0] | | | | | | | | | NrBytesToTransfer[14:0] (1 kB for full-speed) | | | | | | | | | | | | | | | [1] | | V |

[1] Reserved.

[2] EndPt[0].

**Table 85. Start and complete split for isochronous: bit description**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| **DW7** | | | | |
| 63 to 40 | reserved | - | - | - |
| 39 to 32 | ISO_IN_7[7:0] | **HW —** writes | - | **Isochronous IN 7**: Bytes received during μSOF7, if μSA[7] is set to 1 and frame number is correct. |
| **DW6** | | | | |
| 31 to 24 | ISO_IN_6[7:0] | **HW —** writes | - | **Isochronous IN 6**: Bytes received during μSOF6, if μSA[6] is set to 1 and frame number is correct. |
| 23 to 16 | ISO_IN_5[7:0] | **HW —** writes | - | **Isochronous IN 5**: Bytes received during μSOF5, if μSA[5] is set to 1 and frame number is correct. |
| 15 to 8 | ISO_IN_4[7:0] | **HW —** writes | - | **Isochronous IN 4**: Bytes received during μSOF4, if μSA[4] is set to 1 and frame number is correct. |
| 7 to 0 | ISO_IN_3[7:0] | **HW —** writes | - | **Isochronous IN 3**: Bytes received during μSOF3, if μSA[3] is set to 1 and frame number is correct. |
| **DW5** | | | | |
| 63 to 56 | ISO_IN_2[7:0] | **HW —** writes | - | **Isochronous IN 2**: Bytes received during μSOF2, if μSA[2] is set to 1 and frame number is correct. |
| 55 to 48 | ISO_IN_1[7:0] | **HW —** writes | - | **Isochronous IN 1**: Bytes received during μSOF1, if μSA[1] is set to 1 and frame number is correct. |
| 47 to 40 | ISO_IN_0[7:0] | **HW —** writes | - | **Isochronous IN 0**: Bytes received during μSOF0 if μSA[0] is set to 1 and frame number is correct. |
| 39 to 32 | μSCS[7:0] | **SW —** writes (0 → 1)<br>**HW —** writes (1 → 0)<br>After processing | - | **μSCS**: All bits can be set to one for every transfer. It specifies which μSOF the complete split needs to be sent. Valid only for IN. Start split and complete split active bits, μSA = 0000 0001 and μSCS = 0000 0100, will cause SS to execute in μFrame0 and CS in μFrame2. |
| **DW4** | | | | |
| 31 to 29 | Status7[2:0] | **HW —** writes | - | **Status 7**: Isochronous IN or OUT status of μSOF7 |
| 28 to 26 | Status6[2:0] | **HW —** writes | - | **Status 6**: Isochronous IN or OUT status of μSOF6 |
| 25 to 23 | Status5[2:0] | **HW —** writes | - | **Status 5**: Isochronous IN or OUT status of μSOF5 |
| 22 to 20 | Status4[2:0] | **HW —** writes | - | **Status 4**: Isochronous IN or OUT status of μSOF4 |
| 19 to 17 | Status3[2:0] | **HW —** writes | - | **Status 3**: Isochronous IN or OUT status of μSOF3 |
| 16 to 14 | Status2[2:0] | **HW —** writes | - | **Status 2**: Isochronous IN or OUT status of μSOF2 |
| 13 to 11 | Status1[2:0] | **HW —** writes | - | **Status 1**: Isochronous IN or OUT status of μSOF1 |
| 10 to 8 | Status0[2:0] | **HW —** writes | - | **Status 0**: Isochronous IN or OUT status of μSOF0<br>**Bit 0 —** Transaction error (IN and OUT)<br>**Bit 1 —** Babble (IN token only)<br>**Bit 2 —** Underrun (OUT token only) |
| 7 to 0 | μSA[7:0] | **SW —** writes (0 → 1)<br>**HW —** writes (1 → 0)<br>After processing | - | **μSA**: Specifies which μSOF the start split needs to be placed.<br>**For OUT token**: When the frame number of bits DW2[7:3] matches the frame number of the USB bus, these bits are checked for one before they are sent for the μSOF.<br>**For IN token**: Only μSOF0, μSOF1, μSOF2, or μSOF3 can be set to 1. Nothing can be set for μSOF4 and above. |

**Table 85. Start and complete split for isochronous: bit description** *...continued*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| **DW3** | | | | |
| 63 | A | **SW —** sets<br>**HW —** resets | - | **Active**: Write the same value as that in V. |
| 62 | H | **HW —** writes | - | **Halt**: The Halt bit is set when any microframe transfer status has a stalled or halted condition. |
| 61 | B | **HW —** writes | - | **Babble**: This bit corresponds to bit 1 of Status0 to Status7 for every microframe transfer status. |
| 60 | X | **HW —** writes | - | **Transaction error**: This bit corresponds to bit 0 of Status0 to Status7 for every microframe transfer status. |
| 59 | SC | **SW —** writes 0<br>**HW —** updates | - | **Start/complete**:<br>**0 —** Start split<br>**1 —** Complete split |
| 58 | reserved | - | - | - |
| 57 | DT | **HW —** writes<br>**SW —** writes | - | **Data toggle**: Set the Data Toggle bit to start for the PTD. |
| 56 to 44 | reserved | - | - | - |
| 43 to 32 | NrBytes Transferred [11:0] | **HW —** writes | - | **Number of bytes transferred**: This field indicates the number of bytes sent or received for this transaction. |
| **DW2** | | | | |
| 31 to 24 | reserved | - | - | - |
| 23 to 8 | DataStart Address[15:0] | **SW —** writes | - | **Data start address**: This is the start address for data that will be sent on or received from the USB bus. This is the internal memory address and not the CPU address. |
| 7 to 0 | μFrame[7:0] | **SW —** writes | - | **μFrame**: Bits 7 to 3 determine which frame to execute. |
| **DW1** | | | | |
| 63 to 57 | HubAddress [6:0] | **SW —** writes | - | **Hub address**: This indicates the hub address. |
| 56 to 50 | PortNumber [6:0] | **SW —** writes | - | **Port number**: This indicates the port number of the hub or embedded TT. |
| 49 to 47 | reserved | - | - | - |
| 46 | S | **SW —** writes | - | **Split**: This bit indicates whether a split transaction must be executed:<br>**0 —** High-speed transaction<br>**1 —** Split transaction |
| 45 to 44 | EPType[1:0] | **SW —** writes | - | **Transaction type**:<br>**01 —** Isochronous |
| 43 to 42 | Token[1:0] | **SW —** writes | - | **Token**: Token PID for this transaction:<br>**00 —** OUT<br>**01 —** IN |
| 41 to 35 | Device Address[6:0] | **SW —** writes | - | **Device address**: This is the USB address of the function containing the endpoint that is referred to by this buffer. |
| 34 to 32 | EndPt[3:1] | **SW —** writes | - | **Endpoint**: This is the USB address of the endpoint within the function. |

**Table 85. Start and complete split for isochronous: bit description** *…continued*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| **DW0** | | | | |
| 31 | EndPt[0] | **SW —** writes | - | **Endpoint**: This is the USB address of the endpoint within the function. |
| 30 to 29 | reserved | - | - | - |
| 28 to 18 | TT_MPS_Len [10:0] | **SW —** writes | - | **Transaction translator maximum packet size length**: This field indicates the maximum number of bytes that can be sent per start split depending on the number of total bytes needed. If the total bytes to be sent for the entire millisecond is greater than 188 bytes, this field must be set to 188 bytes for an OUT token and 192 bytes for an IN token. Otherwise, this field must be equal to the total bytes sent. |
| 17 to 3 | NrBytesTo Transfer[14:0] | **SW —** writes | - | **Number of bytes to transfer**: This field indicates the number of bytes that can be transferred by this data structure. It is used to indicate the depth of the data field. This field is restricted to 1023 bytes because in siTD the maximum allowable payload for a full-speed device is 1023 bytes. This field indirectly becomes the maximum packet size of the downstream device. |
| 2 to 1 | reserved | - | - | - |
| 0 | V | **SW —** sets<br>**HW —** resets | - | **Valid**:<br>**0 —** This bit is deactivated when the entire PTD is executed, or when a fatal error is encountered.<br>**1 —** The software updates to one when there is payload to be sent or received. The current PTD is active. |

### 10.4.6 Start and complete split for interrupt

Table 86 shows the bit allocation of start and complete split for interrupt.

**Table 86. Start and complete split for interrupt: bit allocation**

| Bit | 63 | 62 | 61 | 60 | 59 | 58 | 57 | 56 | 55 | 54 | 53 | 52 | 51 | 50 | 49 | 48 | 47 | 46 | 45 | 44 | 43 | 42 | 41 | 40 | 39 | 38 | 37 | 36 | 35 | 34 | 33 | 32 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DW7 | reserved | | | | | | | | | | | | | | | | | | | | | | | | INT_IN_7[7:0] | | | | | | | |
| DW5 | INT_IN_2[7:0] | | | | | | | | INT_IN_1[7:0] | | | | | | | | INT_IN_0[7:0] | | | | | | | | µSCS[7:0] | | | | | | | |
| DW3 | A | H | B | X | SC | [1] | DT | Cerr [1:0] | reserved | | | | | | | | | | | | NrBytesTransferred[11:0] (4 kB for full-speed and low-speed) | | | | | | | | | | | |
| DW1 | HubAddress[6:0]` | | | | | | | PortNumber[6:0] | | | | | | | SE[1:0] | | - | S | EP Type [1:0] | | Token [1:0] | | DeviceAddress[6:0] | | | | | | | EndPt[3:0] | | |
| **Bit** | **31** | **30** | **29** | **28** | **27** | **26** | **25** | **24** | **23** | **22** | **21** | **20** | **19** | **18** | **17** | **16** | **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| DW6 | INT_IN_6[7:0] | | | | | | | | INT_IN_5[7:0] | | | | | | | | INT_IN_4[7:0] | | | | | | | | INT_IN_3[7:0] | | | | | | | |
| DW4 | Status7[2:0] | | | Status6[2:0] | | | Status5[2:0] | | Status4[2:0] | | | Status3[2:0] | | | Status2[2:0] | | | Status1[2:0] | | | Status0[2:0] | | | µSA[7:0] | | | | | | | |
| DW2 | reserved | | | | | | | | DataStartAddress[15:0] | | | | | | | | | | | | | | | | µFrame[7:0] (full-speed and low-speed) | | | | | | | |
| DW0 | [2] | | [1] | | | | | MaxPacketLength[10:0] | | | | | | | NrBytesToTransfer[14:0] (4 kB for full-speed and low-speed) | | | | | | | | | | | | | | [1] | | V |

[1] Reserved.

[2] EndPt[0].

**Table 87. Start and complete split for interrupt: bit description**

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| **DW7** | | | | |
| 63 to 40 | reserved | - | - | - |
| 39 to 32 | INT_IN_7[7:0] | **HW —** writes | - | **Interrupt IN 7**: Bytes received during μSOF7, if μSA[7] is set to 1 and frame number is correct. The new value continuously overwrites the old value. |
| **DW6** | | | | |
| 31 to 24 | INT_IN_6[7:0] | **HW —** writes | - | **Interrupt IN 6**: Bytes received during μSOF6, if μSA[6] is set to 1 and frame number is correct. The new value continuously overwrites the old value. |
| 23 to 16 | INT_IN_5[7:0] | **HW —** writes | - | **Interrupt IN 5**: Bytes received during μSOF5, if μSA[5] is set to 1 and frame number is correct. The new value continuously overwrites the old value. |
| 15 to 8 | INT_IN_4[7:0] | **HW —** writes | - | **Interrupt IN 4**: Bytes received during μSOF4, if μSA[4] is set to 1 and frame number is correct. The new value continuously overwrites the old value. |
| 7 to 0 | INT_IN_3[7:0] | **HW —** writes | - | **Interrupt IN 3**: Bytes received during μSOF3, if μSA[3] is set to 1 and frame number is correct. The new value continuously overwrites the old value. |
| **DW5** | | | | |
| 63 to 56 | INT_IN_2[7:0] | **HW —** writes | - | **Interrupt IN 2**: Bytes received during μSOF2, if μSA[2] is set to 1 and frame number is correct. The new value continuously overwrites the old value. |
| 55 to 48 | INT_IN_1[7:0] | **HW —** writes | - | **Interrupt IN 1**: Bytes received during μSOF1, if μSA[1] is set to 1 and frame number is correct. The new value continuously overwrites the old value. |
| 47 to 40 | INT_IN_0[7:0] | **HW —** writes | - | **Interrupt IN 0**: Bytes received during μSOF0, if μSA[0] is set to 1 and frame number is correct. The new value continuously overwrites the old value. |
| 39 to 32 | μSCS[7:0] | **SW —** writes (0 → 1)<br>**HW —** writes (1 → 0)<br>After processing | - | μ**SCS**: All bits can be set to one for every transfer. It specifies which μSOF the complete split needs to be sent. Valid only for IN. Start split and complete split active bits, μSA = 0000 0001 and μSCS = 0000 0100, will cause SS to execute in μFrame0 and CS in μFrame2. |
| **DW4** | | | | |
| 31 to 29 | Status7[2:0] | **HW —** writes | - | **Status 7**: Interrupt IN or OUT status of μSOF7 |
| 28 to 26 | Status6[2:0] | **HW —** writes | - | **Status 6**: Interrupt IN or OUT status of μSOF6 |
| 25 to 23 | Status5[2:0] | **HW —** writes | - | **Status 5**: Interrupt IN or OUT status of μSOF5 |
| 22 to 20 | Status4[2:0] | **HW —** writes | - | **Status 4**: Interrupt IN or OUT status of μSOF4 |
| 19 to 17 | Status3[2:0] | **HW —** writes | - | **Status 3**: Interrupt IN or OUT status of μSOF3 |
| 16 to 14 | Status2[2:0] | **HW —** writes | - | **Status 2**: Interrupt IN or OUT status of μSOF2 |
| 13 to 11 | Status1[2:0] | **HW —** writes | - | **Status 1**: Interrupt IN or OUT status of μSOF1 |
| 10 to 8 | Status0[2:0] | **HW —** writes | - | **Status 0**: Interrupt IN or OUT status of μSOF0<br>**Bit 0 —** Transaction error (IN and OUT)<br>**Bit 1 —** Babble (IN token only)<br>**Bit 2 —** Underrun (OUT token only) |

**Table 87. Start and complete split for interrupt: bit description** *...continued*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 7 to 0 | μSA[7:0] | **SW —** writes (0 → 1)<br>**HW —** writes (1 → 0)<br>After processing | - | **μSA**: Specifies which μSOF the start split needs to be placed.<br>**For OUT token**: When the frame number of bits DW2[7:3] matches the frame number of the USB bus, these bits are checked for one before they are sent for μSOF.<br>**For IN token**: Only μSOF0, μSOF1, μSOF2, or μSOF3 can be set to 1. Nothing can be set for μSOF4 and above. |
| **DW3** | | | | |
| 63 | A | **SW —** sets<br>**HW —** resets | - | **Active**: Write the same value as that in V. |
| 62 | H | **HW —** writes | - | **Halt**: The Halt bit is set when any microframe transfer status has a stalled or halted condition. |
| 61 | B | **HW —** writes | - | **Babble**: This bit corresponds to bit 1 of Status0 to Status7 for every microframe transfer status. |
| 60 | X | **HW —** writes | - | **Transaction error**: This bit corresponds to bit 0 of Status0 to Status7 for every microframe transfer status. |
| 59 | SC | **SW —** writes 0<br>**HW —** updates | - | **Start/complete**:<br>**0 —** Start split<br>**1 —** Complete split |
| 58 | reserved | - | - | - |
| 57 | DT | **HW —** writes<br>**SW —** writes | - | **Data toggle**: For an interrupt transfer, set correct bit to start the PTD. |
| 56 to 55 | Cerr[1:0] | **HW —** writes<br>**SW —** writes | - | **Error counter**: This field corresponds to the Cerr[1:0] field in TD.<br>**00 —** The transaction will not retry.<br>**11 —** The transaction will retry three times. The hardware will decrement these values. |
| 54 to 44 | reserved | - | - | - |
| 43 to 32 | NrBytes Transferred[11:0] | **HW —** writes | - | **Number of bytes transferred**: This field indicates the number of bytes sent or received for this transaction. |
| **DW2** | | | | |
| 31 to 24 | reserved | - | - | - |
| 23 to 8 | DataStart Address[15:0] | **SW —** writes | - | **Data start address**: This is the start address for data that will be sent on or received from the USB bus. This is the internal memory address and not the CPU address. |
| 7 to 0 | μFrame[7:0] | **SW —** writes | - | **μFrame**: Bits 7 to 3 is the polling rate in milliseconds. Polling rate is defined as $2^{(b-1)}$ μSOF; where b = 4 to 16. Executed every millisecond when b is 4. See Table 88. |
| **DW1** | | | | |
| 63 to 57 | HubAddress[6:0] | **SW —** writes | - | **Hub address**: This indicates the hub address. |
| 56 to 50 | PortNumber[6:0] | **SW —** writes | - | **Port number**: This indicates the port number of the hub or embedded TT. |
| 49 to 48 | SE[1:0] | **SW —** writes | - | This depends on the endpoint type and direction. It is valid only for split transactions. Table 89 applies to start split and complete split only. |
| 47 | reserved | - | - | - |

**Table 87. Start and complete split for interrupt: bit description** *...continued*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 46 | S | **SW —** writes | - | **Split**: This bit indicates whether a split transaction has to be executed:<br>**0 —** High-speed transaction<br>**1 —** Split transaction |
| 45 to 44 | EPType[1:0] | **SW —** writes | - | **Transaction type**:<br>**11 —** Interrupt |
| 43 to 42 | Token[1:0] | **SW —** writes | - | **Token**: Token PID for this transaction:<br>**00 —** OUT<br>**01 —** IN |
| 41 to 35 | DeviceAddress [6:0] | **SW —** writes | - | **Device address**: This is the USB address of the function containing the endpoint that is referred to by this buffer. |
| 34 to 32 | EndPt[3:1] | **SW —** writes | - | **Endpoint**: This is the USB address of the endpoint within the function. |
| **DW0** | | | | |
| 31 | EndPt[0] | **SW —** writes | - | **Endpoint**: This is the USB address of the endpoint within the function. |
| 30 to 29 | reserved | - | - | - |
| 28 to 18 | MaxPacket Length[10:0] | **SW —** writes | - | **Maximum packet length**: This field indicates the maximum number of bytes that can be sent to or received from an endpoint in a single data packet. The maximum packet size for the full-speed and low-speed devices is 64 bytes as defined in *Universal Serial Bus Specification Rev. 2.0*. |
| 17 to 3 | NrBytesTo Transfer[14:0] | **SW —** writes | - | **Number of bytes to transfer**: This field indicates the number of bytes that can be transferred by this data structure. It is used to indicate the depth of the data field. The maximum total number of bytes for this transaction is 4 kB. |
| 2 to 1 | reserved | - | - | - |
| 0 | V | **SW —** sets<br>**HW —** resets | - | **Valid**:<br>**0 —** This bit is deactivated when the entire PTD is executed, or when a fatal error is encountered.<br>**1 —** The software updates to one when there is payload to be sent or received. The current PTD is active. |

**Table 88. Microframe description**

| b | Rate | μFrame[7:3] |
|---|---|---|
| 5 | 2 ms | 0 0001 |
| 6 | 4 ms | 0 0010 or 0 0011 |
| 7 | 8 ms | 0 0100 or 0 0111 |
| 8 | 16 ms | 0 1000 or 0 1111 |
| 9 | 32 ms | 1 0000 or 1 1111 |

**Table 89. SE description**

| Interrupt | SE1 | SE0 | Remark |
|---|---|---|---|
| I/O | 1 | 0 | low-speed |
| I/O | 0 | 0 | full-speed |

# 11. Peripheral controller-specific registers

Table 90. Peripheral controller-specific register overview

| Address | Register | Reset value | Reference |
|---|---|---|---|
| **Initialization registers** | | | |
| 00h | Address | 00h | Section 11.2.1 on page 85 |
| 0Ch | Mode | 0000h | Section 11.2.2 on page 85 |
| 10h | Interrupt Configuration | FCh | Section 11.2.3 on page 86 |
| 14h | DcInterruptEnable | 0000 0000h | Section 11.2.4 on page 87 |
| **Data flow registers** | | | |
| 2Ch | Endpoint Index | 20h | Section 11.3.1 on page 89 |
| 28h | Control Function | 00h | Section 11.3.2 on page 90 |
| 20h | Data Port | 0000h | Section 11.3.3 on page 91 |
| 1Ch | Buffer Length | 0000h | Section 11.3.4 on page 92 |
| 1Eh | DcBufferStatus | 00h | Section 11.3.5 on page 93 |
| 04h | Endpoint MaxPacketSize | 0000h | Section 11.3.6 on page 93 |
| 08h | Endpoint Type | 0000h | Section 11.3.7 on page 94 |
| **DMA registers** | | | |
| 30h | DMA Command | FFh | Section 11.4.1 on page 96 |
| 34h | DMA Transfer Counter | 0000 0000h | Section 11.4.2 on page 97 |
| 38h | DcDMAConfiguration | 0001h | Section 11.4.3 on page 98 |
| 3Ch | DMA Hardware | 04h | Section 11.4.4 on page 99 |
| 50h | DMA Interrupt Reason | 0000h | Section 11.4.5 on page 100 |
| 54h | DMA Interrupt Enable | 0000h | Section 11.4.6 on page 101 |
| 58h | DMA Endpoint | 00h | Section 11.4.7 on page 102 |
| 64h | DMA Burst Counter | 0004h | Section 11.4.8 on page 102 |
| **General registers** | | | |
| 18h | DcInterrupt | 0000 0000h | Section 11.5.1 on page 103 |
| 70h | Chip ID | 0017 6320h | Section 11.5.2 on page 105 |
| 74h | Frame Number | 0000h | Section 11.5.3 on page 105 |
| 78h | Scratch | 0000h | Section 11.5.4 on page 105 |
| 7Ch | Unlock Device | - | Section 11.5.5 on page 106 |
| 80h | Interrupt Pulse Width | 1Eh | Section 11.5.6 on page 106 |
| 84h | Test Mode | 00h | Section 11.5.7 on page 107 |

## 11.1 Register access

Register access depends on the bus width used. The ISP1763A uses an 8-bit or 16-bit bus access. For single-byte registers, the upper byte (MSByte) must be ignored.

Endpoint-specific registers are indexed using the Endpoint Index register. The target endpoint must be selected before accessing the following registers:

- Control Function
- Data Port

- Buffer Length
- DcBufferStatus
- Endpoint MaxPacketSize
- Endpoint Type

**Remark:** All reserved bits are not implemented. The bus and bus reset values are not defined. Therefore, writing to these reserved bits will have no effect.

## 11.2 Initialization registers

### 11.2.1 Address register

This register sets the USB assigned address and enables the USB peripheral. Table 91 shows the bit allocation of the register.

The DEVADDR[6:0] bits will be cleared whenever a bus reset, a power-on reset, or a soft reset occurs. The DEVEN bit will be cleared whenever a power-on reset or a soft reset occurs, and will remain unchanged on a bus reset.

In response to standard USB request SET_ADDRESS, the firmware must write the (enabled) peripheral address to the Address register, followed by sending an empty packet to the host. The new peripheral address is activated when the peripheral receives acknowledgment from the host for the empty packet token.

**Table 91. ADDR - Address register (address 00h) bit allocation**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Symbol | DEVEN | DEVADDR[6:0] | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bus reset | unchanged | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

**Table 92. ADDR - Address register (address 00h) bit description**

| Bit | Symbol | Description |
|---|---|---|
| 7 | DEVEN | **Device enable**: Logic 1 enables the device. The device will not respond to the host, unless this bit is set. |
| 6 to 0 | DEVADDR[6:0] | **Device address**: This field specifies the USB device peripheral. |

### 11.2.2 Mode register

This register consists of 2 bytes (bit allocation: see Table 93).

The Mode register controls resume, suspend and wake-up behavior, interrupt activity, soft reset, and clock signals.

**Table 93. MODE - Mode register (address 0Ch) bit allocation**

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Symbol | reserved[1] | | | | | | DMACLK ON | VBUSSTAT |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bus reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Symbol | CLKAON | SNDRSU | GOSUSP | SFRESET | GLINTENA | WKUPCS | reserved[1] | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bus reset | 0 | 0 | 0 | 0 | unchanged | 0 | 0 | unchanged |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

[1] The reserved bits should always be written with the reset value.

**Table 94. MODE - Mode register (address 0Ch) bit description**

| Bit | Symbol | Description |
|---|---|---|
| 15 to 10 | - | reserved |
| 9 | DMACLKON | **DMA clock on**:<br>**1 —** Supply clock to the DMA circuit.<br>**0 —** Power saving mode. The DMA circuit will stop completely to save power. |
| 8 | VBUSSTAT | **$V_{BUS}$ status**: This bit reflects the $V_{BUS}$ pin status when the OTG_DISABLE bit is logic 1. |
| 7 | CLKAON | **Clock always on**:<br>**1 —** Enable the Clock-Always-On feature.<br>**0 —** Disable the Clock-Always-On feature.<br>When the Clock-Always-On feature is disabled, a GOSUSP event can stop the clock. The clock is stopped after a delay of approximately 2 ms. Therefore, the peripheral controller will consume less power.<br>If the Clock-Always-On feature is enabled, clocks are always running and the GOSUSP event is unable to stop the clock while the peripheral controller enters the suspend state. |
| 6 | SNDRSU | **Send resume**: Writing logic 1, followed by logic 0 will generate an upstream resume signal of 10 ms duration, after a 5 ms delay. |
| 5 | GOSUSP | **Go suspend**: Writing logic 1, followed by logic 0 will activate suspend mode. |
| 4 | SFRESET | **Soft reset**: Writing logic 1, followed by logic 0 will enable a software-initiated reset to the ISP1763A. A soft reset is similar to a hardware-initiated reset using the RESET_N pin. |
| 3 | GLINTENA | **Global interrupt enable**: Logic 1 enables all interrupts. Individual interrupts can be masked by clearing the corresponding bits in the DcInterruptEnable register.<br>When this bit is not set, an unmasked interrupt will not generate an interrupt trigger on the interrupt pin. If the global interrupt, however, is enabled while there is any pending unmasked interrupt, an interrupt signal will immediately be generated on the interrupt pin. If the interrupt is set to pulse mode, the interrupt events that were generated before the global interrupt is enabled may be dropped. |
| 2 | WKUPCS | **Wake up on chip select**: Logic 1 enables wake-up through a valid register read on the ISP1763A. A read will invoke the chip clock to restart. A write to the register before the clock is stable may cause malfunctioning. |
| 1 to 0 | - | reserved |

### 11.2.3 Interrupt Configuration register

This 1 byte register determines the behavior and polarity of the INT output. The bit allocation is shown in Table 95. When the USB SIE receives or generates an ACK, NAK, or NYET, it will generate interrupts depending on three Debug mode fields.

**CDBGMOD[1:0] —** Interrupts for control endpoint 0

**DDBGMODIN[1:0] —** Interrupts for data IN endpoints 1 to 7

**DDBGMODOUT[1:0] —** Interrupts for data OUT endpoints 1 to 7

The Debug mode settings for CDBGMOD, DDBGMODIN, and DDBGMODOUT allow you to individually configure when the ISP1763A sends an interrupt to the external microprocessor. Table 97 lists available combinations.

Bit INTPOL controls the signal polarity of the INT output: active HIGH or LOW, rising or falling edge. For level-triggering, bit INTLVL must be made logic 0. By setting INTLVL to logic 1, an interrupt will generate a pulse of 60 ns (edge-triggering).

**Table 95. INTR_CONF - Interrupt Configuration register (address 10h) bit allocation**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Symbol | CDBGMOD[1:0] | | DDBGMODIN[1:0] | | DDBGMODOUT[1:0] | | INTLVL | INTPOL |
| Reset | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 |
| Bus reset | 1 | 1 | 1 | 1 | 1 | 1 | unchanged | unchanged |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

**Table 96. INTR_CONF - Interrupt Configuration register (address 10h) bit description**

| Bit | Symbol | Description |
|---|---|---|
| 7 to 6 | CDBGMOD[1:0] | **Control 0 debug mode**: For values, see Table 97. |
| 5 to 4 | DDBGMODIN[1:0] | **Data debug mode IN**: For values, see Table 97. |
| 3 to 2 | DDBGMODOUT[1:0] | **Data debug mode OUT**: For values, see Table 97. |
| 1 | INTLVL | **Interrupt level**: Selects signaling mode on output INT: 0 = level; 1 = pulsed. In pulsed mode, an interrupt produces a 60 ns pulse. Bus reset value: unchanged.<br>**Remark:** Value written to this bit must be the same as the value written to bit INTR_LEVEL in the HW Mode Control register (address B6h). |
| 0 | INTPOL | **Interrupt polarity**: Selects the signal polarity on output INT; 0 = active LOW; 1 = active HIGH. Bus reset value: unchanged.<br>**Remark:** Value written to this bit must be the same as the value written to bit INTR_POL in the HW Mode Control register (address B6h). |

**Table 97. Debug mode settings**

| Value | CDBGMOD | DDBGMODIN | DDBGMODOUT |
|---|---|---|---|
| 00h | interrupt on all ACK and NAK | interrupt on all ACK and NAK | interrupt on all ACK, NYET, and NAK |
| 01h | interrupt on all ACK | interrupt on ACK | interrupt on ACK and NYET |
| 1Xh | interrupt on all ACK and first NAK[1] | interrupt on all ACK and first NAK[1] | interrupt on all ACK, NYET, and first NAK[1] |

[1] First NAK: The first NAK on an IN or OUT token after a previous ACK response.

### 11.2.4 DcInterruptEnable register

This register enables or disables individual interrupt sources. The interrupt for each endpoint can individually be controlled through the associated IEPnRX or IEPnTX bits, here n represents the endpoint number. All interrupts can globally be disabled through bit GLINTENA in the Mode register (see Table 93).

An interrupt is generated when the USB SIE receives or generates an ACK or NAK on the USB bus. The interrupt generation depends on Debug mode settings of bit fields CDBGMOD[1:0], DDBGMODIN[1:0], and DDBGMODOUT[1:0].

All data IN transactions use Transmit buffers (TX) that are handled by DDBGMODIN bits. All data OUT transactions go through Receive buffers (RX) that are handled by DDBGMODOUT bits. Transactions on control endpoint 0 (IN, OUT, and SETUP) are handled by CDBGMOD bits.

Interrupts caused by events on the USB bus (SOF, suspend, resume, bus reset, set up, and high-speed status) can also be individually controlled. A bus reset disables all enabled interrupts, except bit IEBRST (bus reset) that remains unchanged.

The DcInterruptEnable register consists of 4 bytes. The bit allocation is given in Table 98.

**Table 98. DcInterruptEnable - Device Controller Interrupt Enable register (address 14h) bit allocation**

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Symbol | reserved[1] | | | | | | IEP7TX | IEP7RX |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bus reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| **Bit** | **23** | **22** | **21** | **20** | **19** | **18** | **17** | **16** |
| Symbol | IEP6TX | IEP6RX | IEP5TX | IEP5RX | IEP4TX | IEP4RX | IEP3TX | IEP3RX |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bus reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| **Bit** | **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** |
| Symbol | IEP2TX | IEP2RX | IEP1TX | IEP1RX | IEP0TX | IEP0RX | reserved[1] | IEP0SETUP |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | - | 0 |
| Bus reset | 0 | 0 | 0 | 0 | 0 | 0 | - | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| Symbol | IEVBUS | IEDMA | IEHS_STA | IERESM | IESUSP | IEPSOF | IESOF | IEBRST |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bus reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | unchanged |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

[1] The reserved bits should always be written with the reset value.

**Table 99. DcInterruptEnable - Device Controller Interrupt Enable register (address 14h) bit description**

| Bit | Symbol | Description |
|---|---|---|
| 31 to 26 | - | reserved |
| 25 | IEP7TX | **Interrupt enable endpoint 7 transmit**: Logic 1 enables interrupt from the indicated endpoint. |
| 24 | IEP7RX | **Interrupt enable endpoint 7 receive**: Logic 1 enables interrupt from the indicated endpoint. |
| 23 | IEP6TX | **Interrupt enable endpoint 6 transmit**: Logic 1 enables interrupt from the indicated endpoint. |
| 22 | IEP6RX | **Interrupt enable endpoint 6 receive**: Logic 1 enables interrupt from the indicated endpoint. |
| 21 | IEP5TX | **Interrupt enable endpoint 5 transmit**: Logic 1 enables interrupt from the indicated endpoint. |
| 20 | IEP5RX | **Interrupt enable endpoint 5 receive**: Logic 1 enables interrupt from the indicated endpoint. |
| 19 | IEP4TX | **Interrupt enable endpoint 4 transmit**: Logic 1 enables interrupt from the indicated endpoint. |
| 18 | IEP4RX | **Interrupt enable endpoint 4 receive**: Logic 1 enables interrupt from the indicated endpoint. |

Table 99. DcInterruptEnable - Device Controller Interrupt Enable register (address 14h) bit description *...continued*

| Bit | Symbol | Description |
|---|---|---|
| 17 | IEP3TX | **Interrupt enable endpoint 3 transmit**: Logic 1 enables interrupt from the indicated endpoint. |
| 16 | IEP3RX | **Interrupt enable endpoint 3 receive**: Logic 1 enables interrupt from the indicated endpoint. |
| 15 | IEP2TX | **Interrupt enable endpoint 2 transmit**: Logic 1 enables interrupt from the indicated endpoint. |
| 14 | IEP2RX | **Interrupt enable endpoint 2 receive**: Logic 1 enables interrupt from the indicated endpoint. |
| 13 | IEP1TX | **Interrupt enable endpoint 1 transmit**: Logic 1 enables interrupt from the indicated endpoint. |
| 12 | IEP1RX | **Interrupt enable endpoint 1 receive**: Logic 1 enables interrupt from the indicated endpoint. |
| 11 | IEP0TX | **Interrupt enable endpoint 0 transmit**: Logic 1 enables interrupt from the control IN endpoint 0. |
| 10 | IEP0RX | **Interrupt enable endpoint 0 receive**: Logic 1 enables interrupt from the control OUT endpoint 0. |
| 9 | - | reserved |
| 8 | IEP0SETUP | **Interrupt enable endpoint 0 set-up**: Logic 1 enables interrupt for the set-up data received on endpoint 0. |
| 7 | IEVBUS | **Interrupt enable $V_{BUS}$**: Logic 1 enables interrupt when there is a polarity change on $V_{BUS}$. |
| 6 | IEDMA | **Interrupt enable DMA**: Logic 1 enables interrupt on detecting a DMA status change. |
| 5 | IEHS_STA | **Interrupt enable high-speed status**: Logic 1 enables interrupt on detecting a high-speed status change. |
| 4 | IERESM | **Interrupt enable resume**: Logic 1 enables interrupt on detecting a resume state. |
| 3 | IESUSP | **Interrupt enable suspend**: Logic 1 enables interrupt on detecting a suspend state. |
| 2 | IEPSOF | **Interrupt enable pseudo SOF**: Logic 1 enables interrupt on detecting a pseudo SOF. |
| 1 | IESOF | **Interrupt enable SOF**: Logic 1 enables interrupt on detecting an SOF. |
| 0 | IEBRST | **Interrupt enable bus reset**: Logic 1 enables interrupt on detecting a bus reset. |

## 11.3 Data flow registers

### 11.3.1 Endpoint Index register

The Endpoint Index register selects a target endpoint for register access by the microcontroller. The register consists of 1 byte, and the bit allocation is shown in Table 100.

The following registers are indexed:

- Control Function
- Data Port
- Buffer Length
- DcBufferStatus
- Endpoint MaxPacketSize
- Endpoint Type

For example, to access the OUT data buffer of endpoint 1 using the Data Port register, the Endpoint Index register must be written first with 02h.

**Remark:** The Endpoint Index register and the DMA Endpoint register must not point to the same endpoint, irrespective of IN and OUT.

**Remark:** After writing to the Endpoint Index register, wait for 400 ns before accessing any register.

**Table 100. ENDP_INDEX - Endpoint Index register (address 2Ch) bit allocation**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Symbol | reserved[1] | | EP0SETUP | reserved[1] | ENDPIDX[2:0] | | | DIR |
| Reset | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| Bus reset | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

[1] The reserved bits should always be written with the reset value.

**Table 101. ENDP_INDEX - Endpoint Index register (address 2Ch) bit description**

| Bit | Symbol | Description |
|---|---|---|
| 7 to 6 | - | reserved |
| 5 | EP0SETUP | **Endpoint 0 set up**: Selects the set-up token buffer for endpoint 0. Must be logic 0 for access to endpoints other than the set-up token buffer.<br>**0 —** Any buffer other than the set-up token buffer. This means control IN endpoint, control OUT endpoint, data IN endpoint, and data OUT endpoint.<br>**1 —** Set-up token buffer |
| 4 | - | reserved |
| 3 to 1 | ENDPIDX[2:0] | **Endpoint index**: Selects the target endpoint buffer. |
| 0 | DIR | **Direction**: Sets the target endpoint as IN or OUT.<br>**0 —** Target endpoint refers to OUT (RX) FIFO.<br>**1 —** Target endpoint refers to IN (TX) FIFO. |

**Table 102. Addressing of endpoint buffers**

| Buffer name | EP0SETUP | ENDPIDX | DIR |
|---|---|---|---|
| SETUP | 1 | 00h | 0 |
| Control OUT | 0 | 00h | 0 |
| Control IN | 0 | 00h | 1 |
| Data OUT | 0 | 0Xh | 0 |
| Data IN | 0 | 0Xh | 1 |

### 11.3.2 Control Function register

The Control Function register performs the buffer management on endpoints. It consists of 1 byte, and the bit allocation is given in Table 103. The register bits can stall, clear, or validate any enabled data endpoint. Before accessing this register, the Endpoint Index register must be written first to specify the target endpoint.

**Table 103. CTRL_FUNC- Control Function register (address 28h) bit allocation**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Symbol | reserved[1] | | | CLBUF | VENDP | DSEN | STATUS | STALL |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bus reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

[1] The reserved bits should always be written with the reset value.

Table 104. CTRL_FUNC- Control Function register (address 28h) bit description

| Bit | Symbol | Description |
|---|---|---|
| 7 to 5 | - | reserved |
| 4 | CLBUF | **Clear buffer**: Logic 1 clears the RX buffer of the indexed endpoint. The RX buffer is automatically cleared once the endpoint is completely read. This bit is set only when it is necessary to forcefully clear the buffer.<br>**Remark:** If using a double buffer, to clear both the buffers, issue the CLBUF command two times, that is, set and clear this bit two times. |
| 3 | VENDP | **Validate endpoint**: Logic 1 validates data in the TX FIFO of an IN endpoint for sending on the next IN token. In general, the endpoint is automatically validated when its FIFO byte count has reached endpoint MaxPacketSize. This bit is set only when it is necessary to validate the endpoint with the FIFO byte count that is below the endpoint MaxPacketSize. |
| 2 | DSEN | **Data stage enable**: This bit controls the response of the ISP1763A to a control transfer. After the completion of the set-up stage, the firmware must determine whether a data stage is required. For control OUT, the firmware will set this bit and the ISP1763A goes into the data stage. Otherwise, the ISP1763A will NAK the data stage transfer. For control IN, the firmware will set this bit before writing data to the TX FIFO and validate the endpoint. If no data stage is required, the firmware can immediately set the STATUS bit after the set-up stage. |
| 1 | STATUS | **Status acknowledge**: Only applicable for control IN and OUT.<br>This bit controls the generation of ACK or NAK during the status stage of a SETUP transfer. It is automatically cleared when the status stage is completed and a SETUP token is received. No interrupt signal will be generated.<br>**0 —** Sends NAK<br>**1 —** Sends an empty packet following the IN token (peripheral-to-host) or ACK following the OUT token (host-to-peripheral)<br>**Remark:** The STATUS bit is cleared to zero once the zero-length packet is acknowledged by the device or the PC host.<br>**Remark:** Data transfers preceding the status stage must first be fully completed before the STATUS bit can be set. |
| 0 | STALL | **Stall endpoint**: Logic 1 stalls the indexed endpoint. This bit is not applicable for isochronous transfers.<br>**Remark:** Stalling a data endpoint will confuse the Data Toggle bit about the stalled endpoint because the internal logic picks up from where it is stalled. Therefore, the Data Toggle bit must be reset by disabling and re-enabling the corresponding endpoint (by setting bit ENABLE to logic 0, followed by logic 1 in the Endpoint Type register) to reset the PID. |

### 11.3.3 Data Port register

This 2-byte register provides direct access for a microcontroller to the FIFO of the indexed endpoint. The bit allocation is shown in Table 105.

**Peripheral to host (IN endpoint)**: After each write, an internal counter is automatically incremented, by two in 16-bit mode and one in 8-bit mode, to the next location in the TX FIFO. When all bytes have been written (FIFO byte count = endpoint MaxPacketSize), the buffer is automatically validated. The data packet will then be sent on the next IN token. Whenever required, the Control Function register (bit VENDP) can validate the endpoint whose byte count is less than MaxPacketSize.

**Remark:** The buffer can be automatically validated or cleared by using the Buffer Length register (see Table 106).

**Host to peripheral (OUT endpoint)**: After each read, an internal counter is automatically decremented, by two in 16-bit mode and one in 8-bit mode, to the next location in the RX FIFO. When all bytes have been read, the buffer contents are automatically cleared. A

new data packet can then be received on the next OUT token. Buffer contents can also be cleared through the Control Function register (bit CLBUF), whenever it is necessary to forcefully clear contents.

The Data Port register description when the ISP1763A is in 16-bit mode is given in Table 105.

**Table 105. DATA_PORT - Data Port register (address 20h) bit description**
*Legend: * reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 15 to 0 | DATAPORT [15:0] | R/W | 0000h* | **Data port**: A 500 ns delay starting from the reception of the endpoint interrupt may be required for the first read from the data port. |

### 11.3.4 Buffer Length register

This register determines the current packet size (DATACOUNT) of the indexed endpoint FIFO. The bit allocation is given in Table 106.

The Buffer Length register will be updated with the FIFO size register value whenever there is a write access to the Endpoint MaxPacketSize register (see Table 109). A smaller value can be written when required. After a bus reset, the Buffer Length register is made zero.

**IN endpoint**: When the data transfer is performed in multiples of MaxPacketSize, the Buffer Length register is not significant. This register is useful only when transferring data that is not a multiple of MaxPacketSize. The following two examples demonstrate the significance of the Buffer Length register.

Example 1: Consider that the transfer size is 512 bytes and MaxPacketSize is programmed as 64 bytes, the Buffer Length register need not be filled. This is because the transfer size is a multiple of MaxPacketSize, and MaxPacketSize packets will be automatically validated because the last packet is also of MaxPacketSize.

Example 2: Consider that the transfer size is 510 bytes and MaxPacketSize is programmed as 64 bytes, the Buffer Length register should be filled with 62 bytes just before the microcontroller writes the last packet of 62 bytes. This ensures that the last packet, which is a short packet of 62 bytes, is automatically validated.

Use the VENDP bit in the Control register if you are not using the Buffer Length register.

This is applicable only to the PIO mode access.

**OUT endpoint**: The DATACOUNT value is automatically initialized to the number of data bytes sent by the host on each ACK.

**Remark:** When using a 16-bit microprocessor bus, the last byte of an odd-sized packet is output as the lower byte (LSByte).

**Table 106. BUFFER_LENGTH - Buffer Length register (address 1Ch) bit description**
*Legend: * reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 15 to 0 | DATACOUNT [15:0] | R/W | 0000h* | **Data count**: Determines the current packet size of the indexed endpoint FIFO. |

### 11.3.5 DcBufferStatus register

The endpoint index must first be set before accessing this register for the corresponding endpoint. It reflects the status of the endpoint FIFO. Table 107 shows the bit allocation of the DcBufferStatus register.

**Remark:** Buffer 1 is filled first before filling up buffer 0.

**Remark:** This register is not applicable to the control endpoint.

**Remark:** For the endpoint IN data transfer, the firmware must ensure a 200 ns delay between writing of the data packet and reading the DcBufferStatus register. For the endpoint OUT data transfer, the firmware must also ensure a 200 ns delay between the reception of the endpoint interrupt and reading the DcBufferStatus register.

**Table 107. DcBufferStatus - Device Controller Buffer Status register (address 1Eh) bit allocation**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Symbol | reserved | | | | | | BUF1 | BUF0 |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bus reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R | R | R | R | R | R | R | R |

**Table 108. DcBufferStatus - Device Controller Buffer Status register (address 1Eh) bit description**

| Bit | Symbol | Description |
|---|---|---|
| 7 to 2 | - | reserved |
| 1 to 0 | BUF[1:0] | **Buffer**:<br>**00 —** The buffers are not filled.<br>**01 —** One of the buffers is filled.<br>**10 —** One of the buffers is filled.<br>**11 —** Both the buffers are filled. |

### 11.3.6 Endpoint MaxPacketSize register

This register determines the maximum packet size for all endpoints, except control endpoint 0. The register contains 2 bytes, and the bit allocation is given in Table 109.

Each time the register is written, the Buffer Length registers of the corresponding endpoint is re-initialized to the FFOSZ field value. NTRANS bits control the number of transactions allowed in a single microframe for high-speed isochronous and interrupt endpoints only.

**Table 109. ENDP_MAXPKTSIZE - Endpoint MaxPacketSize register (address 04h) bit allocation**

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Symbol | reserved[1] | | | NTRANS[1:0] | | FFOSZ[10:8] | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bus reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| Symbol | FFOSZ[7:0] | | | | | | | |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bus reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

[1] The reserved bits should always be written with the reset value.

**Table 110. ENDP_MAXPKTSIZE - Endpoint MaxPacketSize register (address 04h) bit description**

| Bit | Symbol | Description |
|---|---|---|
| 15 to 13 | - | reserved |
| 12 to 11 | NTRANS[1:0] | **Number of transactions**: HS mode only.<br>**00 —** One packet per microframe.<br>**01 —** Two packets per microframe.<br>**10 —** Three packets per microframe.<br>**11 —** reserved<br>These bits are applicable only for isochronous or interrupt transactions. |
| 10 to 0 | FFOSZ[10:0] | **FIFO size**: Sets the FIFO size in bytes for the indexed endpoint. Applies to both high-speed and full-speed operations (see Table 111). |

The ISP1763A supports all the transfers given in *Universal Serial Bus Specification Rev. 2.0*.

Each programmable FIFO can be independently configured using its Endpoint MaxPacketSize register (R/W: 04h), but the total physical size of all enabled endpoints (IN plus OUT), including the set-up token buffer, control IN, and control OUT, must not exceed 4096 bytes.

### 11.3.7 Endpoint Type register

This register sets the endpoint type of the indexed endpoint: isochronous, bulk, or interrupt. It also serves to enable the endpoint and configure it for double buffering. Automatic generation of an empty packet for a zero-length TX buffer can be disabled using bit NOEMPKT. The register contains 2 bytes. See Table 111.

**Table 111. ENDP_TYPE - Endpoint Type register (address 08h) bit allocation**

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Symbol | reserved[1] | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bus reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| Symbol | reserved[1] | | | NOEMPKT | ENABLE | DBLBUF | ENDPTYP[1:0] | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bus reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

[1] The reserved bits should always be written with the reset value.

Table 112. ENDP_TYPE - Endpoint Type register (address 08h) bit description

| Bit | Symbol | Description |
|---|---|---|
| 15 to 5 | - | reserved |
| 4 | NOEMPKT | **No empty packet**: Logic 0 causes the ISP1763A to return a null length packet for the IN token after the DMA IN transfer is complete. Set to logic 1 to disable the generation of the null length packet. |
| 3 | ENABLE | **Endpoint enable**: Logic 1 enables the FIFO of the indexed endpoint. The memory size is allocated as specified in the Endpoint MaxPacketSize register. Logic 0 disables the FIFO.<br>**Remark:** Stalling a data endpoint will confuse the Data Toggle bit on the stalled endpoint because the internal logic picks up from where it has stalled. Therefore, the Data Toggle bit must be reset by disabling and re-enabling the corresponding endpoint (by setting bit ENABLE to logic 0, followed by logic 1 in the Endpoint Type register) to reset the PID. |
| 2 | DBLBUF | **Double buffering**: Logic 1 enables double buffering for the indexed endpoint. Logic 0 disables double buffering. |
| 1 to 0 | ENDPTYP [1:0] | **Endpoint type**: These bits select the endpoint type as follows.<br>**00 —** Not used<br>**01 —** Isochronous<br>**10 —** Bulk<br>**11 —** Interrupt |

## 11.4 DMA registers

The Generic DMA (GDMA) transfer can be done by writing the proper opcode in the DMA Command register. The control bits are given in Table 113.

**GDMA read/write (opcode = 00h/01h) for Generic DMA slave mode**

The GDMA (slave) can operate in counter mode. RD_N/DS_N/RE_N/OE_N and WR_N/RW_N/WE_N are DMA data strobe signals. These signals are also used as data strobe signals during the PIO access. An internal multiplex will redirect these signals to the DMA controller for the DMA transfer or to registers for the PIO access.

In counter mode, the DIS_XFER_CNT bit in the DcDMAConfiguration register must be set to logic 0. The DMA Transfer Counter register must be programmed before any DMA command is issued. The DMA transfer counter is set by writing from the LSByte to the MSByte (address: 34h to 37h). The internal DMA transfer counter is updated when the MSByte is written. Once the DMA transfer is started, the transfer counter starts decrementing and on reaching 0, the DMA_XFER_OK bit is set and an interrupt is generated by the ISP1763A.

The DMA transfer starts once the DMA command is issued. Any of the following three ways will terminate this DMA transfer:

- Detecting an internal EOT (short packet on an OUT token)
- Resetting the DMA
- GDMA stop command

There are two interrupts that are programmable to differentiate the method of DMA termination: the INT_EOT and DMA_XFER_OK bits in the DMA Interrupt Reason register. For details, see Table 123.

**Table 113. Control bits for GDMA read/write (opcode = 00h/01h)**

| Control bits | Description | Reference |
|---|---|---|
| **DcDMAConfiguration register** | | |
| MODE[1:0] | Determines the active read or write data strobe signals | Table 119 |
| WIDTH | Selects the DMA bus width: 16 or 8 bits | |
| DIS_XFER_CNT | Disables the use of the DMA Transfer Counter | |
| **DMA Hardware register** | | |
| ACK_POL, DREQ_POL | Select the polarity of the DMA handshake signals | Table 122 |

**Remark:** The DMA bus defaults to 3-state, until a DMA command is executed. All the other control signals are not 3-stated.

### 11.4.1 DMA Command register

The DMA Command register is a 1-byte register (for bit allocation, see Table 114) that initiates all DMA transfer activities on the DMA controller. The register is write-only; reading it will return FFh.

**Remark:** The DMA bus will be in 3-state until a DMA command is executed.

**Table 114. DMA_CMD - DMA Command register (address 30h) bit allocation**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| **Symbol** | DMA_CMD[7:0] | | | | | | | |
| **Reset** | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| **Bus reset** | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| **Access** | W | W | W | W | W | W | W | W |

**Table 115. DMA_CMD - DMA Command register (address 30h) bit description**

| Bit | Symbol | Description |
|---|---|---|
| 7 to 0 | DMA_CMD[7:0] | DMA command code; see Table 116. |

**Table 116. DMA commands**

| Code | Name | Description |
|---|---|---|
| 00h | GDMA Read | **Generic DMA IN token transfer**: Data is transferred from the external DMA bus to the internal buffer. |
| 01h | GDMA Write | **Generic DMA OUT token transfer**: Data is transferred from the internal buffer to the external DMA bus. |
| 02h to 0Dh | - | reserved |
| 0Eh | Validate Buffer | **Validate buffer**: Request from the microcontroller to validate the endpoint buffer, following a DMA-to-USB data transfer. |
| 0Fh | Clear Buffer | **Clear buffer**: Request from the microcontroller to clear the endpoint buffer, after a DMA-to-USB data transfer. It clears the TX buffer of the indexed endpoint; the RX buffer is not affected. The TX buffer is automatically cleared once data is sent on the USB bus. This command is issued only when it is necessary to forcefully clear the buffer.<br>**Remark:** If using a double buffer, to clear both the buffers, issue the Clear Buffer command two times. |
| 10h | - | reserved |

Table 116. DMA commands *...continued*

| Code | Name | Description |
|---|---|---|
| 11h | Reset DMA | **Reset DMA**: Initializes the DMA core to its power-on reset state.<br>**Remark:** When the DMA core is reset during the Reset DMA command, the DREQ, DACK, RD_N/DS_N/RE_N/OE_N, and WR_N/RW_N/WE_N handshake pins will temporarily be asserted. This can confuse the external DMA controller. To prevent this, start the external DMA controller **only after** the DMA reset. |
| 12h | - | reserved |
| 13h | GDMA Stop | **GDMA stop**: This command stops the GDMA data transfer. Any data in the OUT endpoint that is not transferred by the DMA will remain in the buffer. The FIFO data for the IN endpoint will be written to the endpoint buffer. An interrupt bit will be set to indicate that the DMA Stop command is complete. |
| 14h to FFh | - | reserved |

### 11.4.2 DMA Transfer Counter register

This 4-byte register sets up the total byte count for a DMA transfer (DMACR). It indicates the remaining number of bytes left for transfer. The bit allocation is given in Table 117.

**For IN endpoint —** Because there is a FIFO in the ISP1763A DMA controller, some data may remain in the FIFO during the DMA transfer. The maximum FIFO size is 8 bytes, and the maximum delay time for data to be shifted to endpoint buffer is 60 ns.

**For OUT endpoint —** Data will not be cleared for the endpoint buffer until all the data has been read from the DMA FIFO.

Table 117. DMA_XFR_CTR - DMA Transfer Counter register (address 34h) bit allocation

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Symbol | DMACR4 = DMACR[31:24] | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bus reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| **Bit** | **23** | **22** | **21** | **20** | **19** | **18** | **17** | **16** |
| Symbol | DMACR3 = DMACR[23:16] | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bus reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| **Bit** | **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** |
| Symbol | DMACR2 = DMACR[15:8] | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bus reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| Symbol | DMACR1 = DMACR[7:0] | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bus reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

**Table 118. DMA_XFR_CTR - DMA Transfer Counter register (address 34h) bit description**

| Bit | Symbol | Description |
|---|---|---|
| 31 to 24 | DMACR4, DMACR[31:24] | **DMA counter 4**: DMA transfer counter byte 4 (MSB) |
| 23 to 16 | DMACR3, DMACR[23:16] | **DMA counter 3**: DMA transfer counter byte 3 |
| 15 to 8 | DMACR2, DMACR[15:8] | **DMA counter 2**: DMA transfer counter byte 2 |
| 7 to 0 | DMACR1, DMACR[7:0] | **DMA counter 1**: DMA transfer counter byte 1 (LSB) |

### 11.4.3 DcDMAConfiguration register

This register defines the DMA configuration for GDMA mode. The DcDMAConfiguration register consists of 2 bytes. The bit allocation is given in Table 119.

**Table 119. DcDMAConfiguration - Device Controller Direct Memory Access Configuration register (address 38h) bit allocation**

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Symbol | reserved[1] | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bus reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| Symbol | DIS_XFER_CNT | reserved[1] | | | MODE[1:0] | | reserved[1] | WIDTH |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| Bus reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

[1] The reserved bits should always be written with the reset value.

**Table 120. DcDMAConfiguration - Device Controller Direct Memory Access Configuration register (address 38h) bit description**

| Bit[1] | Symbol | Description |
|---|---|---|
| 15 to 8 | - | reserved |
| 7 | DIS_XFER_CNT | **Disable transfer counter**: Write logic 0 to perform DMA operation. Logic 1 disables the DMA transfer counter (see Table 117). |
| 6 to 4 | - | reserved |
| 3 to 2 | MODE[1:0] | **Mode**:<br>**00 —** WR_N/RW_N/WE_N slave strobes data from the DMA bus into the ISP1763A; RD_N/DS_N/RE_N/OE_N slave puts data from the ISP1763A on the DMA bus<br>**01, 10, 11 —** reserved |
| 1 | - | reserved |
| 0 | WIDTH | **Width**: This bit selects the DMA bus width for the GDMA slave.<br>**0 —** 8-bit data bus<br>**1 —** 16-bit data bus<br>**Remark:** Both this bit and bit DATA_BUS_WIDTH must be configured with the same bus width, for example, 16 bits or 8 bits. |

[1] The DREQ pin will be driven only after you perform a write access to the DcDMAConfiguration register (that is, after you have configured the DcDMAConfiguration register).

### 11.4.4 DMA Hardware register

This register defines the DMA configuration for GDMA mode. The bit allocation is given in Table 121.

**Table 121. DMA_HW - DMA Hardware register (address 3Ch) bit allocation**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Symbol | reserved[1] | | | | DACK_POL | DREQ_POL | reserved[1] | |
| Reset | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| Bus reset | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

[1] The reserved bits should always be written with the reset value.

**Table 122. DMA_HW - DMA Hardware register (address 3Ch) bit description**

| Bit | Symbol | Description |
|---|---|---|
| 7 to 4 | | reserved |
| 3 | DACK_POL | **DACK polarity**: Selects the DMA acknowledgment polarity.<br>**0 —** DACK is active LOW.<br>**1 —** DACK is active HIGH.<br>**Remark:** Value written to this bit must be the same as the value written to bit DACK_POL in the HW Mode Control register (address B6h). |
| 2 | DREQ_POL | **DREQ polarity**: Selects the DMA request polarity.<br>**0 —** DREQ is active LOW.<br>**1 —** DREQ is active HIGH.<br>**Remark:** Value written to this bit must be the same as the value written to bit DREQ_POL in the HW Mode Control register (address B6h). |
| 1 to 0 | - | reserved |

### 11.4.5 DMA Interrupt Reason register

This 2-byte register shows the source(s) of the DMA interrupt. Each bit is refreshed after a DMA command is executed. An interrupt source is cleared by writing logic 1 to the corresponding bit. On detecting the interrupt, the external microprocessor must read the DMA Interrupt Reason register and mask it with the corresponding bits in the DMA Interrupt Enable register to determine the source of the interrupt.

The bit allocation is given in Table 123.

**Table 123. DMA_INTR_REASON - DMA Interrupt Reason register (address 50h) bit allocation**

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Symbol | reserved[1] | | | GDMA_STOP | reserved[1] | INT_EOT | reserved[1] | DMA_XFER_OK |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bus reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Symbol | reserved[1] | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bus reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

[1] The reserved bits should always be written with the reset value.

**Table 124. DMA_INTR_REASON - DMA Interrupt Reason register (address 50h) bit description**

| Bit | Symbol | Description |
|---|---|---|
| 15 to 13 | - | reserved |
| 12 | GDMA_STOP | **GDMA stop**: When the GDMA_STOP command is issued to the DMA Command register, it means that the DMA transfer has successfully terminated. |
| 11 | - | reserved |
| 10 | INT_EOT | **Internal EOT**: Logic 1 indicates that an internal EOT is detected; see Table 125. |
| 9 | - | reserved |
| 8 | DMA_XFER_OK | **DMA transfer OK**: Logic 1 indicates that the DMA transfer has been completed, that is, DMA transfer counter has become zero. |
| 7 to 0 | - | reserved |

**Table 125. Internal EOT-functional relation with bit DMA_XFER_OK**

| INT_EOT | DMA_XFER_OK | Description |
|---|---|---|
| 1 | 0 | During the DMA transfer, there is a premature termination with a short packet. |
| 1 | 1 | DMA transfer is completed with a short packet. The DMA transfer counter has reached 0. |
| 0 | 1 | DMA transfer is completed without any short packet. The DMA transfer counter has reached 0. |

Table 126 shows the status of the bits in the DMA Interrupt Reason register when the corresponding bits in the DcInterrupt register are set.

**Table 126. Status of the bits in the DMA Interrupt Reason register**

| Status | INT_EOT[1] | DMA_XFER_OK[1] | |
|---|---|---|---|
| | | Counter enabled | Counter disabled |
| IN full | 0 | 1 | 0 |
| IN short | 0 | 1 | 0 |
| OUT full | 0 | 1 | 0 |
| OUT short | 1[2] | 1 | 0 |

[1] 1 indicates that the bit is set and 0 indicates that the bit is not set. A bit is set when the corresponding EOT condition is met.

[2] The value of INT_EOT may not be accurate if an external or internal transfer counter is programmed with a value that is lower than the transfer that the host requests. To terminate an OUT transfer with INT_EOT, the external or internal DMA counter should be programmed as a multiple of the full-packet length of the DMA endpoint. When a short packet is successfully transferred by DMA, INT_EOT is set.

### 11.4.6 DMA Interrupt Enable register

This 2-byte register controls the interrupt generation of the source bits in the DMA Interrupt Reason register. The bit allocation is given in Table 127.

Logic 1 enables the interrupt generation. The values after a (bus) reset are logic 0 (disabled).

**Table 127. DMA_INTR_EN - DMA Interrupt Enable register (address 54h) bit allocation**

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Symbol | reserved[1] | | | IE_GDMA_STOP | reserved[1] | IE_INT_EOT | reserved[1] | IE_DMA_XFER_OK |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bus reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| Symbol | reserved[1] | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bus reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

[1] The reserved bits should always be written with the reset value.

**Table 128. DMA_INTR_EN - DMA Interrupt Enable register (address 54h) bit description**

| Bit | Symbol | Description |
|---|---|---|
| 15 to 13 | - | reserved |
| 12 | IE_GDMA_STOP | **Interrupt enable GDMA stop**: Logic 1 enables the GDMA stop interrupt. |
| 11 | - | reserved |
| 10 | IE_INT_EOT | **Interrupt enable internal EOT**: Logic 1 enables the internal EOT interrupt. |
| 9 | - | reserved |
| 8 | IE_DMA_XFER_OK | **Interrupt enable DMA transfer OK**: Logic 1 enables the DMA transfer complete interrupt. |
| 7 to 0 | - | reserved |

### 11.4.7 DMA Endpoint register

This 1-byte register selects a USB endpoint FIFO as a source or destination for DMA transfers. The bit allocation is given in Table 129.

**Table 129. DMA_ENDP - DMA Endpoint register (address 58h) bit allocation**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Symbol | reserved[1] | | | | EPIDX[2:0] | | | DMADIR |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bus reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

[1] The reserved bits should always be written with the reset value.

**Table 130. DMA_ENDP - DMA Endpoint register (address 58h) bit description**

| Bit | Symbol | Description |
|---|---|---|
| 7 to 4 | - | reserved |
| 3 to 1 | EPIDX[2:0] | **Endpoint index**: Selects the indicated endpoint for the DMA access |
| 0 | DMADIR | **DMA direction**:<br>**0 —** Selects the RX/OUT FIFO for DMA read transfers.<br>**1 —** Selects the TX/IN FIFO for DMA write transfers. |

The DMA Endpoint register must not reference the endpoint that is indexed by the Endpoint Index register (2Ch) at any time. Doing so will result in data corruption. Therefore, if the DMA Endpoint register is unused, point it to an unused endpoint. If the DMA Endpoint register, however, is pointed to an active endpoint, the firmware must not reference the same endpoint on the Endpoint Index register.

### 11.4.8 DMA Burst Counter register

Table 131 shows the bit allocation of the register.

**Table 131. DMA_BRST_CTR - DMA Burst Counter register (address 64h) bit allocation**

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Symbol | reserved[1] | | | BURSTCOUNTER[12:8] | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bus reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| Symbol | BURSTCOUNTER[7:0] | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| Bus reset | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

[1] The reserved bits should always be written with the reset value.

**Table 132. DMA_BRST_CTR - DMA Burst Counter register (address 64h) bit description**

| Bit | Symbol | Description |
|---|---|---|
| 15 to 13 | - | reserved |
| 12 to 0 | BURST COUNTER [12:0] | **Burst counter**: This register defines the burst length. The counter must be programmed to be a multiple of two in 16-bit mode and one in 8-bit mode.<br>The value of the burst counter must be programmed so that the buffer counter is a factor of the burst counter. DREQ will drop when the number of bytes transferred equals to the burst counter at every DMA read or write cycle. |

## 11.5 General registers

### 11.5.1 DcInterrupt register

The DcInterrupt register consists of 4 bytes. The bit allocation is given in Table 133.

When a bit is set in the DcInterrupt register, it indicates that the hardware condition for an interrupt has occurred. When the DcInterrupt register content is non-zero, the INT output will be asserted. On detecting the interrupt, the external microprocessor must read the DcInterrupt register to determine the source of the interrupt.

Each endpoint buffer has a dedicated interrupt bit (EPnTX, EPnRX). In addition, various bus states can generate an interrupt: resume, suspend, pseudo SOF, SOF, and bus reset. The DMA controller has only one interrupt bit: the source for a DMA interrupt is shown in the DMA Interrupt Reason register.

Each interrupt bit can individually be cleared by writing logic 1. The DMA Interrupt bit can be cleared by writing logic 1 to the related interrupt source bit in the DMA Interrupt Reason register and writing logic 1 to the DMA bit of the DcInterrupt register.

**Table 133. DcInterrupt - Device Controller Interrupt register (address 18h) bit allocation**

| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 |
|---|---|---|---|---|---|---|---|---|
| Symbol | reserved[1] | | | | | | EP7TX | EP7RX |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bus reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| **Bit** | **23** | **22** | **21** | **20** | **19** | **18** | **17** | **16** |
| Symbol | EP6TX | EP6RX | EP5TX | EP5RX | EP4TX | EP4RX | EP3TX | EP3RX |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bus reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| **Bit** | **15** | **14** | **13** | **12** | **11** | **10** | **9** | **8** |
| Symbol | EP2TX | EP2RX | EP1TX | EP1RX | EP0TX | EP0RX | reserved[1] | EP0SETUP |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bus reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| Symbol | VBUS | DMA | HS_STAT | RESUME | SUSP | PSOF | SOF | BRESET |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bus reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | unchanged |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

[1] The reserved bits should always be written with the reset value.

**Table 134. DcInterrupt - Device Controller Interrupt register (address 18h) bit description**

| Bit | Symbol | Description |
|---|---|---|
| 31 to 26 | - | reserved |
| 25 | EP7TX | **Endpoint 7 transmit**: Logic 1 indicates the endpoint 7 TX buffer as the interrupt source. |
| 24 | EP7RX | **Endpoint 7 receive**: Logic 1 indicates the endpoint 7 RX buffer as the interrupt source. |
| 23 | EP6TX | **Endpoint 6 transmit**: Logic 1 indicates the endpoint 6 TX buffer as the interrupt source. |
| 22 | EP6RX | **Endpoint 6 receive**: Logic 1 indicates the endpoint 6 RX buffer as the interrupt source. |
| 21 | EP5TX | **Endpoint 5 transmit**: Logic 1 indicates the endpoint 5 TX buffer as the interrupt source. |
| 20 | EP5RX | **Endpoint 5 receive**: Logic 1 indicates the endpoint 5 RX buffer as the interrupt source. |

**Table 134. DcInterrupt - Device Controller Interrupt register (address 18h) bit description** *...continued*

| Bit | Symbol | Description |
|---|---|---|
| 19 | EP4TX | **Endpoint 4 transmit**: Logic 1 indicates the endpoint 4 TX buffer as the interrupt source. |
| 18 | EP4RX | **Endpoint 4 receive**: Logic 1 indicates the endpoint 4 RX buffer as the interrupt source. |
| 17 | EP3TX | **Endpoint 3 transmit**: Logic 1 indicates the endpoint 3 TX buffer as the interrupt source. |
| 16 | EP3RX | **Endpoint 3 receive**: Logic 1 indicates the endpoint 3 RX buffer as the interrupt source. |
| 15 | EP2TX | **Endpoint 2 transmit**: Logic 1 indicates the endpoint 2 TX buffer as the interrupt source. |
| 14 | EP2RX | **Endpoint 2 receive**: Logic 1 indicates the endpoint 2 RX buffer as the interrupt source. |
| 13 | EP1TX | **Endpoint 1 transmit**: Logic 1 indicates the endpoint 1 TX buffer as the interrupt source. |
| 12 | EP1RX | **Endpoint 1 receive**: Logic 1 indicates the endpoint 1 RX buffer as the interrupt source. |
| 11 | EP0TX | **Endpoint 0 transmit**: Logic 1 indicates the endpoint 0 data TX buffer as the interrupt source. |
| 10 | EP0RX | **Endpoint 0 receive**: Logic 1 indicates the endpoint 0 data RX buffer as the interrupt source. |
| 9 | - | reserved |
| 8 | EP0SETUP | **Endpoint 0 set-up**: Logic 1 indicates that a set-up token was received on endpoint 0. |
| 7 | VBUS | **$V_{BUS}$**: Logic 1 indicates there is a polarity change on $V_{BUS}$. |
| 6 | DMA | **DMA status**: Logic 1 indicates a change in the DMA Interrupt Reason register. |
| 5 | HS_STAT | **High-speed status**: Logic 1 indicates a change from full-speed to high-speed mode (HS connection). This bit is not set when the system goes into the full-speed suspend. |
| 4 | RESUME | **Resume status**: Logic 1 indicates that a status change from suspend to resume (active) was detected. |
| 3 | SUSP | **Suspend status**: Logic 1 indicates that a status change from active to suspend was detected on the bus. |
| 2 | PSOF | **Pseudo SOF interrupt**: Logic 1 indicates that a pseudo SOF or μSOF was received. Pseudo SOF is an internally generated clock signal (full-speed: 1 ms period, high-speed: 125 μs period) that is not synchronized to the USB bus SOF or μSOF. |
| 1 | SOF | **SOF interrupt**: Logic 1 indicates that a SOF or μSOF was received. |
| 0 | BRESET | **Bus reset**: Logic 1 indicates that a USB bus reset was detected. |

### 11.5.2 Chip ID register

This read-only register contains the chip identification and hardware version numbers. The firmware must check this information to determine functions and features supported. The bit allocation is shown in Table 135.

**Table 135. DcChipID - Device Controller Chip Identifier register (address 70h) bit description**
*Legend: * reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 31 to 0 | CHIPID[31:0] | R | 0017 6320h* | **Chip ID**: This registers represents the hardware version number (20h) and the chip ID (1763h) for the peripheral controller. |

### 11.5.3 Frame Number register

This read-only register contains the frame number of the last successfully received Start-Of-Frame (SOF). The register contains 2 bytes. The bit allocation is given in Table 136.

Table 136. FRAME_NO - Frame Number register (address 74h) bit allocation

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Symbol | reserved | | MICROSOF[2:0] | | | SOFR[10:8] | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bus reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R | R | R | R | R | R | R | R |
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| Symbol | SOFR[7:0] | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bus reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R | R | R | R | R | R | R | R |

Table 137. FRAME_NO - Frame Number register (address 74h) bit description

| Bit | Symbol | Description |
|---|---|---|
| 15 to 14 | - | reserved |
| 13 to 11 | MICROSOF[2:0] | μ**SOF received**: Microframe number of the last successfully received SOF. |
| 10 to 0 | SOFR[10:0] | **SOF received**: Frame number of the last successfully received SOF. |

### 11.5.4 Scratch register

This 16-bit register can be used by the firmware to save and restore information. For example, the device status before it enters the suspend state; see Table 138.

Table 138. SCRATCH - Scratch register (address 78h) bit allocation

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Symbol | SFIRH[7:0] | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bus reset | unchanged | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| Symbol | SFIRL[7:0] | | | | | | | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bus reset | unchanged | | | | | | | |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Table 139. SCRATCH - Scratch register (address 78h) bit description

| Bit | Symbol | Description |
|---|---|---|
| 15 to 8 | SFIRH[7:0] | Scratch firmware information register (higher byte) |
| 7 to 0 | SFIRL[7:0] | Scratch firmware information register (lower byte) |

### 11.5.5 Unlock Device register

To protect registers from getting corrupted when the ISP1763A goes into suspend, the write operation is disabled. In this case, when the chip resumes, the Unlock Device command must first be issued to this register before attempting to write to the rest of the registers. This is done by writing unlock code (AA37h) to this register. The bit allocation of the Unlock Device register is given in Table 140.

**Table 140. UNLOCK_DEV - Unlock Device register (address 7Ch) bit allocation**

| Bit | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 |
|---|---|---|---|---|---|---|---|---|
| Symbol | ULCODE[15:8] = AAh | | | | | | | |
| Reset | not applicable | | | | | | | |
| Bus reset | not applicable | | | | | | | |
| Access | W | W | W | W | W | W | W | W |
| **Bit** | **7** | **6** | **5** | **4** | **3** | **2** | **1** | **0** |
| Symbol | ULCODE[7:0] = 37h | | | | | | | |
| Reset | not applicable | | | | | | | |
| Bus reset | not applicable | | | | | | | |
| Access | W | W | W | W | W | W | W | W |

**Table 141. UNLOCK_DEV - Unlock Device register (address 7Ch) bit description**

| Bit | Symbol | Description |
|---|---|---|
| 15 to 0 | ULCODE[15:0] | **Unlock code**: Writing data AA37h unlocks internal registers and FIFOs for writing, following a resume. |

### 11.5.6 Interrupt Pulse Width register

Table 142 shows the bit description of the register.

**Table 142. INTR_PULSE_WIDTH - Interrupt Pulse Width register (address 80h) bit description**

*Legend: * reset value*

| Bit | Symbol | Access | Value | Description |
|---|---|---|---|---|
| 15 to 0 | INTR_PULSE_ WIDTH[15:0 | R/W | 1Eh* | **Interrupt pulse width**: The interrupt signal pulse width is configurable while it is in pulse signaling mode. The minimum pulse width is 33.3 ns when this register value is 1. The power-on reset value of 1Eh allows a pulse of 1 μs to be generated. |

### 11.5.7 Test Mode register

This 1-byte register allows the firmware to set the DP and DM pins to predetermined states for testing purposes. The bit allocation is given in Table 143.

**Remark:** Only one bit can be set to logic 1 at a time.

**Table 143. TEST_MODE - Test Mode register (address 84h) bit allocation**

| Bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| Symbol | FORCEHS | reserved[1] | | FORCEFS | PRBS | KSTATE | JSTATE | SE0_NAK |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Bus reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Access | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

[1] The reserved bits should always be written with the reset value.

**Table 144. TEST_MODE - Test Mode register (address 84h) bit description**

| Bit | Symbol | Description |
|---|---|---|
| 7 | FORCEHS | **Force high-speed**: Logic 1[1] forces the hardware to high-speed mode only and disables the chirp detection logic. |
| 6 to 5 | - | reserved. |
| 4 | FORCEFS | **Force full-speed**: Logic 1[1] forces the physical layer to full-speed mode only and disables the chirp detection logic. |
| 3 | PRBS | **Predetermined random pattern**: Logic 1[2] sets the DP and DM pins to toggle in a predetermined random pattern. |
| 2 | KSTATE | **K state**: Writing logic 1[2] sets the DP and DM pins to the K state. |
| 1 | JSTATE | **J state**: Writing logic 1[2] sets the DP and DM pins to the J state. |
| 0 | SE0_NAK | **SE0 NAK**: Writing logic 1[2] sets the DP and DM pins to a high-speed quiescent state. The device only responds to a valid high-speed IN token with a NAK. |

[1] Either FORCEHS or FORCEFS must be set at a time.

[2] Of the four bits (PRBS, KSTATE, JSTATE, and SE0_NAK), only one bit must be set at a time.

# 12. Limiting values

**Table 145. Limiting values**

*In accordance with the Absolute Maximum Rating System (IEC 60134).*

| Symbol | Parameter | Conditions | Min | Max | Unit |
|---|---|---|---|---|---|
| $V_{CC(I/O)}$ | input/output supply voltage | | −0.5 | +4.6 | V |
| $V_{CC(3V3)}$ | supply voltage | | −0.5 | +4.6 | V |
| $V_I$ | input voltage | on pin X1/CLKIN | −0.5 | +2.5 | V |
| | | on pins DP1, DM1, DP2, DM2, ID | −0.5 | +4.6 | V |
| | | on pins OC1/$V_{BUS1}$, OC2/$V_{BUS2}$, PSW1_N, PSW2_N | −0.5 | +5.5 | V |
| | | other digital I/O pins | −0.5 | $V_{CC(I/O)}$ + 0.5 | V |
| $I_{lu}$ | latch-up current | $V_I$ < 0 V or $V_I$ > $V_{CC(3V3)}$ | - | 100 | mA |
| $V_{esd}$ | electrostatic discharge voltage | $I_{LI}$ < 1 μA; human body model (JESD22-A114-B) | −2000 | +2000 | V |
| | | $I_{LI}$ < 1 μA; machine model (JESD22-A115-A) | - | +200 | V |
| $T_{stg}$ | storage temperature | | −40 | +125 | °C |

# 13. Recommended operating conditions

**Table 146. Recommended operating conditions**

| Symbol | Parameter | Conditions | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| $V_{CC(I/O)}$ | input/output supply voltage | $V_{CC(I/O)}$ = 3.3 V | 3.0 | 3.3 | 3.6 | V |
| | | $V_{CC(I/O)}$ = 1.8 V | 1.65 | 1.8 | 1.95 | V |
| $V_{CC(3V3)}$ | supply voltage | | 3.0 | - | 3.6 | V |
| $V_I$ | input voltage | on pin X1/CLKIN | 0 | 1.2 | 1.3 | V |
| | | on pins DP1, DM1, DP2, DM2, ID | 0 | - | 3.6 | V |
| | | on pins OC1/$V_{BUS1}$, OC2/$V_{BUS2}$, PSW1_N, PSW2_N | 0 | - | 5.25 | V |
| | | other digital I/O pins | - | - | $V_{CC(I/O)}$ | V |
| $T_{amb}$ | ambient temperature | | −40 | - | +85 | °C |
| $T_j$ | junction temperature | | −40 | - | +125 | °C |

# 14. Static characteristics

**Table 147. Static characteristics: supply pins**

*$V_{CC(3V3)}$ = 3.0 V to 3.6 V; $V_{CC(I/O)}$ = 1.65 V to 1.95 V or 3.0 V to 3.6 V; $T_{amb}$ = −40 °C to +85 °C; unless otherwise specified. Typical case refers to $V_{CC(3V3)}$ = 3.3 V, $V_{CC(I/O)}$ = 3.3 V; $T_{amb}$ = +25 °C; unless otherwise specified.*

| Symbol | Parameter | Conditions | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| $I_{CC(3V3)}$ | supply current | power-down mode[1] | - | 2 | 20 | μA |
| | | deep sleep mode | - | 150 | 350 | μA |
| | | host mode; bus idle | - | 20 | - | mA |
| | | host mode; one port working (high-speed) | - | 36 | - | mA |
| | | host mode; two ports working (high-speed) | - | 55 | - | mA |
| $I_{CC(I/O)}$ | supply current on $V_{CC(I/O)}$ | static; no bus activity | - | - | 20 | μA |

[1] When the I/O pins are in definite state.

**Table 148. Static characteristics: digital pins**

*$V_{CC(3V3)}$ = 3.0 V to 3.6 V; $V_{CC(I/O)}$ = 1.65 V to 1.95 V or 3.0 V to 3.6 V; $T_{amb}$ = −40 °C to +85 °C; unless otherwise specified. Typical case refers to $V_{CC(3V3)}$ = 3.3 V, $V_{CC(I/O)}$ = 3.3 V; $T_{amb}$ = +25 °C; unless otherwise specified.*

| Symbol | Parameter | Conditions | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| $V_{IH}$ | HIGH-level input voltage | | $0.7V_{CC(I/O)}$ | - | - | V |
| $V_{IL}$ | LOW-level input voltage | | - | - | $0.3V_{CC(I/O)}$ | V |
| $V_{hys}$ | hysteresis voltage | | 0.4 | - | 0.7 | V |
| $V_{OL}$ | LOW-level output voltage | $I_{OL}$ = 1.5 mA when $V_{CC(I/O)}$ = 1.65 V to 1.95 V | - | - | 0.4 | V |
| | | $I_{OL}$ = 3.0 mA when $V_{CC(I/O)}$ = 3.0 V to 3.6 V | - | - | 0.4 | V |
| $V_{OH}$ | HIGH-level output voltage | $I_{OH}$ = 3.3 mA when $V_{CC(I/O)}$ = 1.65 V to 1.95 V | $V_{CC(I/O)} - 0.4$ | - | - | V |
| | | $I_{OH}$ = 6.5 mA when $V_{CC(I/O)}$ = 3.0 V to 3.6 V | $V_{CC(I/O)} - 0.4$ | - | - | V |
| $I_{LI}$ | input leakage current | $V_I$ = 0 V to $V_{CC(I/O)}$; push-pull pins | - | - | 1 | μA |
| | | open-drain pins | - | - | 20 | μA |
| $C_{in}$ | input capacitance | | - | - | 3 | pF |

**Table 149. Static characteristics: USB interface block (pins DM1, DM2, DP1, DP2)**

*$V_{CC(3V3)}$ = 3.0 V to 3.6 V; $V_{CC(I/O)}$ = 1.65 V to 1.95 V or 3.0 V to 3.6 V; $T_{amb}$ = −40 °C to +85 °C; unless otherwise specified. Typical case refers to $V_{CC(3V3)}$ = 3.3 V, $V_{CC(I/O)}$ = 3.3 V; $T_{amb}$ = +25 °C; unless otherwise specified.*

| Symbol | Parameter | Conditions | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| **Input levels for high-speed** | | | | | | |
| $V_{HSSQ}$ | high-speed squelch detection threshold voltage (differential signal amplitude) | squelch detected | - | - | 100 | mV |
| | | no squelch detected | 150 | - | - | mV |
| $V_{HSDSC}$ | high-speed disconnect detection threshold voltage (differential signal amplitude) | disconnect detected | 625 | - | - | mV |
| | | disconnect not detected | - | - | 525 | mV |

**Table 149. Static characteristics: USB interface block (pins DM1, DM2, DP1, DP2)** *...continued*

*$V_{CC(3V3)}$ = 3.0 V to 3.6 V; $V_{CC(I/O)}$ = 1.65 V to 1.95 V or 3.0 V to 3.6 V; $T_{amb}$ = −40 °C to +85 °C; unless otherwise specified. Typical case refers to $V_{CC(3V3)}$ = 3.3 V, $V_{CC(I/O)}$ = 3.3 V; $T_{amb}$ = +25 °C; unless otherwise specified.*

| Symbol | Parameter | Conditions | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| $V_{HSCM}$ | high-speed data signaling common mode voltage range (guideline for receiver) | | −50 | - | +500 | mV |
| **Output levels for high-speed** | | | | | | |
| $V_{HSOI}$ | high-speed idle level voltage | | −10 | - | +10 | mV |
| $V_{HSOH}$ | high-speed data signaling HIGH-level voltage | | 360 | - | 440 | mV |
| $V_{HSOL}$ | high-speed data signaling LOW-level voltage | | −10 | - | +10 | mV |
| $V_{CHIRPJ}$ | chirp J level (differential voltage) | | 700[1] | - | 1100 | mV |
| $V_{CHIRPK}$ | chirp K level (differential voltage) | | −900[1] | - | −500 | mV |
| **Input levels for full-speed and low-speed** | | | | | | |
| $V_{IH}$ | HIGH-level input voltage | drive | 2.0 | - | - | V |
| $V_{IHZ}$ | HIGH-level input voltage | floating | 2.7 | - | 3.6 | V |
| $V_{IL}$ | LOW-level input voltage | | - | - | 0.8 | V |
| $V_{DI}$ | differential input sensitivity | $\|V_{DP} - V_{DM}\|$ | 0.2 | - | - | V |
| $V_{CM}$ | differential common mode voltage range | | 0.8 | - | 2.5 | V |
| **Output levels for full-speed and low-speed** | | | | | | |
| $V_{OH}$ | HIGH-level output voltage | | 2.8 | - | 3.6 | V |
| $V_{OL}$ | LOW-level output voltage | | 0 | - | 0.3 | V |
| $V_{CRS}$ | output signal crossover voltage | | 1.3 | - | 2.0 | V |
| **Impedance** | | | | | | |
| $Z_{DRV}$ | driver output impedance | includes the $R_S$ resistor | 40.5 | 45 | 49.5 | Ω |

[1] The HS termination resistor is disabled, and the pull-up resistor is connected. Only during reset, when both the hub and the device are capable of the high-speed operation.

**Table 150. Static characteristics: $V_{BUS}$ comparators**

*$V_{CC(3V3)}$ = 3.0 V to 3.6 V; $V_{CC(I/O)}$ = 1.65 V to 1.95 V or 3.0 V to 3.6 V; $T_{amb}$ = −40 °C to +85 °C; unless otherwise specified. Typical case refers to $V_{CC(3V3)}$ = 3.3 V, $V_{CC(I/O)}$ = 3.3 V; $T_{amb}$ = +25 °C; unless otherwise specified.*

| Symbol | Parameter | Conditions | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| $V_{A\_VBUS\_VLD}$ | A-device $V_{BUS}$ valid voltage | | - | 4.5 | - | V |
| $V_{A\_SESS\_VLD}$ | A-device session valid voltage | | 0.8 | 1.6 | 2.0 | V |
| $V_{hys(A\_SESS\_VLD)}$ | A-device session valid hysteresis voltage | | - | 210 | - | mV |
| $V_{B\_SESS\_VLD}$ | B-device session valid voltage | | 2.0 | 3.6 | 4.0 | V |
| $V_{hys(B\_SESS\_VLD)}$ | B-device session valid hysteresis voltage | | - | 220 | - | mV |
| $V_{B\_SESS\_END}$ | B-device session end voltage | | 0.2 | 0.5 | 0.8 | V |
| $V_{hys(B\_SESS\_END)}$ | B-device session end hysteresis voltage | | - | 160 | - | mV |

**Table 151. Static characteristics: $V_{BUS}$ resistors**

*$V_{CC(3V3)}$ = 3.0 V to 3.6 V; $V_{CC(I/O)}$ = 1.65 V to 1.95 V or 3.0 V to 3.6 V; $T_{amb}$ = −40 °C to +85 °C; unless otherwise specified. Typical case refers to $V_{CC(3V3)}$ = 3.3 V, $V_{CC(I/O)}$ = 3.3 V; $T_{amb}$ = +25 °C; unless otherwise specified.*

| Symbol | Parameter | Conditions | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| $R_{UP(VBUS)}$ | pull-up resistance on pin OC1/$V_{BUS1}$ | connect to $V_{CC(3V3)}$ when VBUS_CHRG = 1 | - | 1.2 | - | kΩ |
| $R_{DN(VBUS)}$ | pull-down resistance on pin OC1/$V_{BUS1}$ | connect to ground when VBUS_DISCHRG = 1 | - | 1.8 | - | kΩ |
| $R_{I(idle)(VBUS)}$ | idle input resistance on pin OC1/$V_{BUS1}$ | | 40[1] | 65 | 100 | kΩ |

[1] The $V_{BUS}$ input impedance may be lower than 40 kΩ for a short period of time when $V_{BUS}$ rises above $V_{CC(3V3)}$.

**Table 152. Static characteristics: ID detection circuit**

*$V_{CC(3V3)}$ = 3.0 V to 3.6 V; $V_{CC(I/O)}$ = 1.65 V to 1.95 V or 3.0 V to 3.6 V; $T_{amb}$ = −40 °C to +85 °C; unless otherwise specified. Typical case refers to $V_{CC(3V3)}$ = 3.3 V, $V_{CC(I/O)}$ = 3.3 V; $T_{amb}$ = +25 °C; unless otherwise specified.*

| Symbol | Parameter | Conditions | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| $V_{th(ID)}$ | ID detector threshold voltage | | 1.0 | - | 2.0 | V |
| $R_{UP(ID)}$ | ID pull-up resistance | bit ID_PULLUP = 1 | - | 20 | - | kΩ |

# 15. Dynamic characteristics

**Table 153. Dynamic characteristics: system clock**

*$V_{CC(3V3)}$ = 3.0 V to 3.6 V; $V_{CC(I/O)}$ = 1.65 V to 1.95 V or 3.0 V to 3.6 V; $T_{amb}$ = −40 °C to +85 °C; unless otherwise specified. Typical case refers to $V_{CC(3V3)}$ = 3.3 V, $V_{CC(I/O)}$ = 3.3 V; $T_{amb}$ = +25 °C; unless otherwise specified.*

| Symbol | Parameter | Conditions | | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|---|
| **Crystal oscillator** | | | | | | | |
| $f_{clk}$ | clock frequency | FREQSEL2 = 0, FREQSEL1 = 0 | [1] | - | 12 | - | MHz |
| | | FREQSEL2 = 0, FREQSEL1 = 1 | [1] | - | 19.2 | - | MHz |
| | | FREQSEL2 = 1, FREQSEL1 = 0 | [1] | - | 24 | - | MHz |
| **External clock input** | | | | | | | |
| J | external clock jitter | | | - | - | 500 | ps |
| δ | clock duty cycle | | | - | 50 | - | % |
| $V_{i(X1/CLKIN)}$ | input voltage on pin X1/CLKIN | | | - | 1.2 | - | V |
| $t_r$ | rise time | | | - | - | 3 | ns |
| $t_f$ | fall time | | | - | - | 3 | ns |

[1] Recommended accuracy of the clock frequency is 50 ppm for the crystal and the oscillator.

**Table 154. Dynamic characteristics: power-up and reset**

*$V_{CC(3V3)}$ = 3.0 V to 3.6 V; $V_{CC(I/O)}$ = 1.65 V to 1.95 V or 3.0 V to 3.6 V; $T_{amb}$ = −40 °C to +85 °C; unless otherwise specified. Typical case refers to $V_{CC(3V3)}$ = 3.3 V, $V_{CC(I/O)}$ = 3.3 V; $T_{amb}$ = +25 °C; unless otherwise specified.*

| Symbol | Parameter | Conditions | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| $t_{DMY\_RD(RESET\_N)}$ | active-LOW reset to dummy read operation | regulator filter capacitor $C_{REG(1V2)}$ = 4.7 μF | 10 | - | | ms |
| $t_{DMY\_RD(RESET)}$ | active-HIGH reset to dummy read operation | regulator filter capacitor $C_{REG(1V2)}$ = 4.7 μF | 0 | - | - | ns |
| $t_{OPR\_RD(DMY\_RD)}$ | dummy read to first valid read operation | regulator filter capacitor $C_{REG(1V2)}$ = 4.7 μF | 10 | - | - | ms |

**Table 155. Dynamic characteristics: digital pins**

*$V_{CC(3V3)}$ = 3.0 V to 3.6 V; $V_{CC(I/O)}$ = 1.65 V to 1.95 V or 3.0 V to 3.6 V; $T_{amb}$ = −40 °C to +85 °C; unless otherwise specified. Typical case refers to $V_{CC(3V3)}$ = 3.3 V, $V_{CC(I/O)}$ = 3.3 V; $T_{amb}$ = +25 °C; unless otherwise specified.*

| Symbol | Parameter | Conditions | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| SR | slew rate | standard load (rise, fall) | 1 | - | 4 | V/ns |

**Table 156. Dynamic characteristics: high-speed source electrical**

*$V_{CC(3V3)}$ = 3.0 V to 3.6 V; $V_{CC(I/O)}$ = 1.65 V to 1.95 V or 3.0 V to 3.6 V; $T_{amb}$ = −40 °C to +85 °C; unless otherwise specified. Typical case refers to $V_{CC(3V3)}$ = 3.3 V, $V_{CC(I/O)}$ = 3.3 V; $T_{amb}$ = +25 °C; unless otherwise specified.*

| Symbol | Parameter | Conditions | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| **Driver characteristics** | | | | | | |
| $t_{HSR}$ | rise time (10% to 90%) | | 500 | - | - | ps |
| $t_{HSF}$ | fall time (10% to 90%) | | 500 | - | - | ps |

**Table 156. Dynamic characteristics: high-speed source electrical** *…continued*

*$V_{CC(3V3)}$ = 3.0 V to 3.6 V; $V_{CC(I/O)}$ = 1.65 V to 1.95 V or 3.0 V to 3.6 V; $T_{amb}$ = −40 °C to +85 °C; unless otherwise specified. Typical case refers to $V_{CC(3V3)}$ = 3.3 V, $V_{CC(I/O)}$ = 3.3 V; $T_{amb}$ = +25 °C; unless otherwise specified.*

| Symbol | Parameter | Conditions | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| **Clock timing** | | | | | | |
| $t_{HSDRAT}$ | high-speed data rate | | 479.76 | - | 480.24 | Mbit/s |
| $t_{HSFRAM}$ | microframe interval | | 124.9375 | - | 125.0625 | µs |
| $t_{HSRFI}$ | consecutive microframe interval difference | | 1 | - | four high-speed bit times | ns |

**Table 157. Dynamic characteristics: full-speed source electrical**

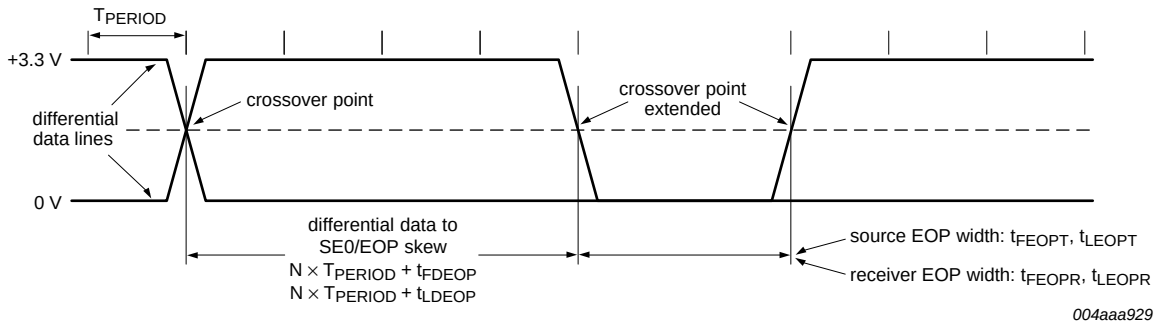
*$V_{CC(3V3)}$ = 3.0 V to 3.6 V; $V_{CC(I/O)}$ = 1.65 V to 1.95 V or 3.0 V to 3.6 V; $T_{amb}$ = −40 °C to +85 °C; unless otherwise specified. Typical case refers to $V_{CC(3V3)}$ = 3.3 V, $V_{CC(I/O)}$ = 3.3 V; $T_{amb}$ = +25 °C; unless otherwise specified.*

| Symbol | Parameter | Conditions | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| **Driver characteristics** | | | | | | |
| $t_{FR}$ | rise time | $C_L$ = 50 pF; 10% to 90% of $\|V_{OH} - V_{OL}\|$ | 4 | - | 20 | ns |
| $t_{FF}$ | fall time | $C_L$ = 50 pF; 90% to 10% of $\|V_{OH} - V_{OL}\|$ | 4 | - | 20 | ns |
| $t_{FRFM}$ | differential rise and fall time matching | | 90 | - | 111.1 | % |
| **Data timing: see Figure 9** | | | | | | |
| $t_{FDEOP}$ | source jitter for differential transition to SE0 transition | full-speed timing | −2 | - | +5 | ns |
| $t_{FEOPT}$ | source SE0 interval of EOP | | 160 | - | 175 | ns |
| $t_{FEOPR}$ | receiver SE0 interval of EOP | | 82 | - | - | ns |
| $t_{LDEOP}$ | upstream facing port source jitter for differential transition to SE0 transition | low-speed timing | −40 | - | +100 | ns |
| $t_{LEOPT}$ | source SE0 interval of EOP | | 1.25 | - | 1.5 | µs |
| $t_{LEOPR}$ | receiver SE0 interval of EOP | | 670 | - | - | ns |
| $t_{FST}$ | width of SE0 interval during differential transition | | - | - | 14 | ns |

**Table 158. Dynamic characteristics: low-speed source electrical**

*$V_{CC(3V3)}$ = 3.0 V to 3.6 V; $V_{CC(I/O)}$ = 1.65 V to 1.95 V or 3.0 V to 3.6 V; $T_{amb}$ = −40 °C to +85 °C; unless otherwise specified. Typical case refers to $V_{CC(3V3)}$ = 3.3 V, $V_{CC(I/O)}$ = 3.3 V; $T_{amb}$ = +25 °C; unless otherwise specified.*

| Symbol | Parameter | Conditions | Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|
| **Driver characteristics** | | | | | | |
| $t_{LR}$ | transition time: rise time | | 75 | - | 300 | ns |
| $t_{LF}$ | transition time: fall time | | 75 | - | 300 | ns |
| $t_{LRFM}$ | rise and fall time matching | | 90 | - | 125 | % |

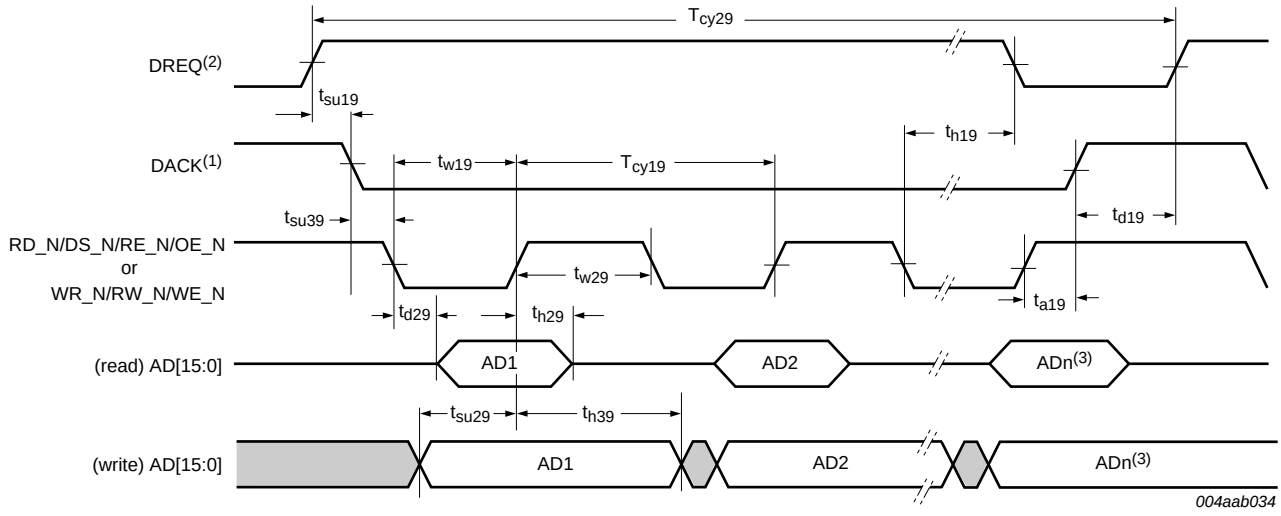



$T_{PERIOD}$ is the bit duration corresponding to the USB data rate.

**Fig 9. USB source differential data-to-EOP transition skew and EOP width**

## 15.1 Timing

### 15.1.1 DMA




DREQ is continuously asserted until the last transfer is done or the FIFO is full or empty.

Data strobes: read and write.

(1) Programmable polarity: shown as active LOW.

(2) Programmable polarity: shown as active HIGH.

(3) In single cycle mode, n = 1, read or write only asserts one time when DREQ asserts.

**Fig 10. DMA read and write**

**Table 159. DMA timing**

*$V_{CC(3V3)}$ = 3.0 V to 3.6 V; $V_{CC(I/O)}$ = 1.65 V to 1.95 V or 3.0 V to 3.6 V; $T_{amb}$ = −40 °C to +85 °C; unless otherwise specified. Typical case refers to $V_{CC(3V3)}$ = 3.3 V, $V_{CC(I/O)}$ = 3.3 V; $T_{amb}$ = +25 °C; unless otherwise specified.*

| Symbol | Parameter description | Host DMA | | Peripheral DMA | | Unit |
|---|---|---|---|---|---|---|
| | | Min | Max | Min | Max | |
| $t_{su19}$ | DREQ set-up time before DACK assertion | 0 | - | 0 | - | ns |
| $t_{d19}$ | DACK deassertion to next DREQ assertion time | 56 | - | 82 | - | ns |
| $t_{h19}$ | DREQ hold time after last strobe assertion | 0 | 53 | 0 | 53 | ns |
| $t_{w19}$ | RD_N/DS_N/RE_N/OE_N or WR_N/RW_N/WE_N pulse width | 25 | - | 40 | - | ns |
| $t_{w29}$ | RD_N/DS_N/RE_N/OE_N or WR_N/RW_N/WE_N recovery time | 21 | - | 35 | - | ns |
| $t_{d29}$ | data valid time after RD_N/DS_N/RE_N/OE_N assertion | - | 30 | - | 30 | ns |
| $t_{h29}$ | read data hold time after RD_N/DS_N/RE_N/OE_N deassertion | 2 | 10 | 2 | 10 | ns |
| $t_{h39}$ | write data hold time after WR_N/RW_N/WE_N deassertion | 0 | - | 0 | - | ns |
| $t_{su29}$ | write data set-up time before WR_N/RW_N/WE_N deassertion | 10 | - | 10 | - | ns |

**Table 159. DMA timing** *...continued*

*$V_{CC(3V3)}$ = 3.0 V to 3.6 V; $V_{CC(I/O)}$ = 1.65 V to 1.95 V or 3.0 V to 3.6 V; $T_{amb}$ = −40 °C to +85 °C; unless otherwise specified. Typical case refers to $V_{CC(3V3)}$ = 3.3 V, $V_{CC(I/O)}$ = 3.3 V; $T_{amb}$ = +25 °C; unless otherwise specified.*

| Symbol | Parameter description | Host DMA | | Peripheral DMA | | Unit |
|---|---|---|---|---|---|---|
| | | Min | Max | Min | Max | |
| $t_{su39}$ | DACK set-up time before RD_N/DS_N/RE_N/OE_N or WR_N/RW_N/WE_N assertion | 0 | - | 0 | - | ns |
| $t_{a19}$ | DACK deassertion after RD_N/DS_N/RE_N/OE_N or WR_N/RW_N/WE_N deassertion | 0 | - | 0 | - | ns |
| $T_{cy19}$ | DMA read/write cycle time | 51 | - | 75 | - | ns |
| $T_{cy29}$ | DMA request cycle time | 104 | - | 104 | - | ns |

### 15.1.2 PIO

**Table 160. PIO timing**

*$V_{CC(3V3)}$ = 3.0 V to 3.6 V; $V_{CC(I/O)}$ = 3.0 V to 3.6 V; $T_{amb}$ = −40 °C to +85 °C; unless otherwise specified. Typical case refers to $V_{CC(3V3)}$ = 3.3 V, $V_{CC(I/O)}$ = 3.3 V; $T_{amb}$ = +25 °C; unless otherwise specified.*

| Symbol | Parameter description | Condition | SRAM mode | | Multiplex mode | | Unit |
|---|---|---|---|---|---|---|---|
| | | | Min | Max | Min | Max | |
| $t_{CS}$ | CS_N/CE_N set-up time before WR_N/RW_N/WE_N or RD_N/DS_N/RE_N/OE_N LOW | | 0 | - | 0 | - | ns |
| $t_{CH}$ | CS_N/CE_N hold time after WR_N/RW_N/WE_N or RD_N/DS_N/RE_N/OE_N HIGH | | 0 | - | 0 | - | ns |
| $t_{CP}$ | CS_N/CE_N pulse width for read | | 30 | - | 30 | - | ns |
| | CS_N/CE_N pulse width for write | | 17 | - | 17 | - | ns |
| $t_{CSADVAL}$ | CS_N/CE_N set-up time before address latch | | - | - | 6 | - | ns |
| $t_{ASRW}$ | address set-up time before WR_N/RW_N/WE_N or RD_N/DS_N/RE_N/OE_N LOW | | 0 | - | - | - | ns |
| $t_{AHRW}$ | address hold time after WR_N/RW_N/WE_N or RD_N/DS_N/RE_N/OE_N LOW | | 10 | - | - | - | ns |
| $t_{AH}$ | address hold time after address latch | | - | - | 0 | - | ns |
| $t_{AS}$ | address set-up time before address latch | | - | - | 10 | - | ns |
| $t_{AP}$ | address latch pulse width | | - | - | 17 | - | ns |
| $t_{WC}$ | write cycle time | host or OTG | 38 | - | 68 | | ns |
| | | peripheral | 100 | - | 100 | | ns |
| $t_{WP}$ | WR_N/RW_N/WE_N pulse width | | 17 | - | 17 | - | ns |
| $t_{DH}$ | RD_N/DS_N/RE_N/OE_N HIGH to output Hi-Z | | 1 | 10 | 1 | 10 | ns |
| | WR_N/RW_N/WE_N HIGH to input Hi-Z | | 0 | - | 0 | - | ns |
| $t_{DADVH}$ | data set-up time before data latch | | 10 | - | 10 | - | ns |
| $t_{OE}$ | RD_N/DS_N/RE_N/OE_N LOW to data output enable | | 2 | - | 2 | - | ns |
| $t_{BDS}$ | ready to WR_N/RW_N/WE_N or RD_N/DS_N/RE_N/OE_N LOW | | - | - | 5 | - | ns |

**Table 160. PIO timing** *...continued*

*$V_{CC(3V3)}$ = 3.0 V to 3.6 V; $V_{CC(I/O)}$ = 3.0 V to 3.6 V; $T_{amb}$ = −40 °C to +85 °C; unless otherwise specified.*
*Typical case refers to $V_{CC(3V3)}$ = 3.3 V, $V_{CC(I/O)}$ = 3.3 V; $T_{amb}$ = +25 °C; unless otherwise specified.*

| Symbol | Parameter description | Condition | SRAM mode | | Multiplex mode | | Unit |
|---|---|---|---|---|---|---|---|
| | | | Min | Max | Min | Max | |
| $t_{RP}$ | RD_N/DS_N/RE_N/OE_N pulse width | | 30 | - | 30 | - | ns |
| $t_{RC}$ | RD_N/DS_N/RE_N/OE_N cycle time | host or OTG | 49 | - | 68 | - | ns |
| | | peripheral | 100 | - | 100 | - | ns |
| $t_{P13}$ | memory read pre-fetch time | | 105 | - | 105 | - | ns |

**Table 161. PIO timing**

*$V_{CC(3V3)}$ = 3.0 V to 3.6 V; $V_{CC(I/O)}$ = 1.65 V to 1.95 V; $T_{amb}$ = −40 °C to +85 °C; unless otherwise specified.*
*Typical case refers to $V_{CC(3V3)}$ = 3.3 V, $V_{CC(I/O)}$ = 1.8 V; $T_{amb}$ = +25 °C; unless otherwise specified.*

| Symbol | Parameter description | Conditions | SRAM mode | | Multiplex mode | | Unit |
|---|---|---|---|---|---|---|---|
| | | | Min | Max | Min | Max | |
| $t_{CS}$ | CS_N/CE_N set-up time before WR_N/RW_N/WE_N or RD_N/DS_N/RE_N/OE_N LOW | | 0 | - | 0 | - | ns |
| $t_{CH}$ | CS_N/CE_N hold time after WR_N/RW_N/WE_N or RD_N/DS_N/RE_N/OE_N HIGH | | 0 | - | 0 | - | ns |
| $t_{CP}$ | CS_N/CE_N pulse width for read | | 38 | - | 38 | - | ns |
| | CS_N/CE_N pulse width for write | | 17 | - | 17 | - | ns |
| $t_{CSADVAL}$ | CS_N/CE_N set-up time before address latch | | - | - | 6 | - | ns |
| $t_{ASRW}$ | address set-up time before WR_N/RW_N/WE_N or RD_N/DS_N/RE_N/OE_N LOW | | 0 | - | - | - | ns |
| $t_{AHRW}$ | address hold time after WR_N/RW_N/WE_N or RD_N/DS_N/RE_N/OE_N LOW | | 10 | - | - | - | ns |
| $t_{AH}$ | address hold time after address latch | | - | - | 0 | - | ns |
| $t_{AS}$ | address set-up time before address latch | | - | - | 10 | - | ns |
| $t_{AP}$ | address latch pulse width | | - | - | 17 | - | ns |
| $t_{WC}$ | write cycle time | host or OTG | 38 | - | 68 | | ns |
| | | peripheral | 100 | - | 100 | | ns |
| $t_{WP}$ | WR_N/RW_N/WE_N pulse width | | 17 | - | 17 | - | ns |
| $t_{DH}$ | RD_N/DS_N/RE_N/OE_N HIGH to output Hi-Z | | 1 | 10 | 1 | 10 | ns |
| | WR_N/RW_N/WE_N HIGH to input Hi-Z | | 0 | - | 0 | - | ns |
| $t_{DADVH}$ | data set-up time before data latch | | 10 | - | 10 | - | ns |
| $t_{OE}$ | RD_N/DS_N/RE_N/OE_N LOW to data output enable | | 2 | - | 2 | - | ns |
| $t_{BDS}$ | ready to WR_N/RW_N/WE_N or RD_N/DS_N/RE_N/OE_N LOW | | - | - | 5 | - | ns |

**Table 161. PIO timing** *...continued*

*$V_{CC(3V3)}$ = 3.0 V to 3.6 V; $V_{CC(I/O)}$ = 1.65 V to 1.95 V; $T_{amb}$ = −40 °C to +85 °C; unless otherwise specified.*
*Typical case refers to $V_{CC(3V3)}$ = 3.3 V, $V_{CC(I/O)}$ = 1.8 V; $T_{amb}$ = +25 °C; unless otherwise specified.*

| Symbol | Parameter description | Conditions | SRAM mode | | Multiplex mode | | Unit |
|---|---|---|---|---|---|---|---|
| | | | Min | Max | Min | Max | |
| $t_{RP}$ | RD_N/DS_N/RE_N/OE_N pulse width | | 38 | - | 38 | - | ns |
| $t_{RC}$ | RD_N/DS_N/RE_N/OE_N cycle time | host or OTG | 54 | - | 68 | - | ns |
| | | peripheral | 100 | - | 100 | - | ns |
| $t_{P13}$ | memory read pre-fetch time | | 113 | - | 113 | - | ns |

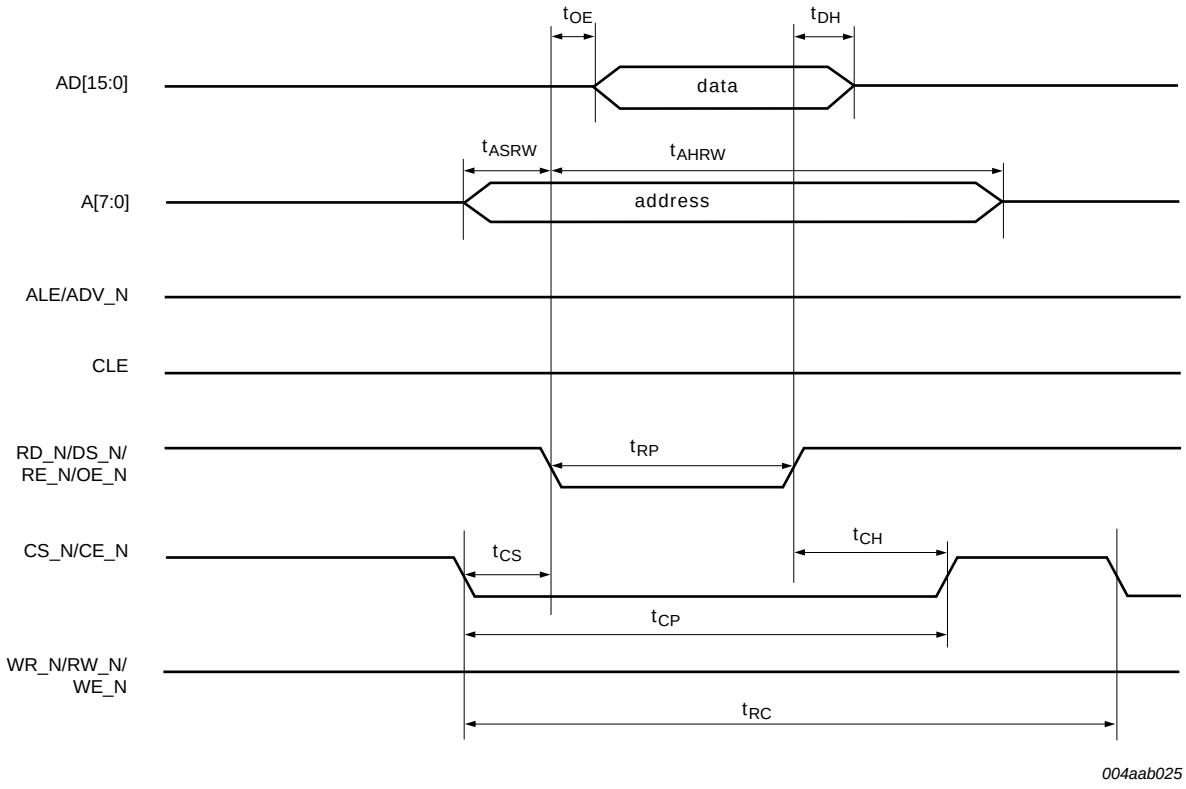



**Fig 11. Read in SRAM mode**

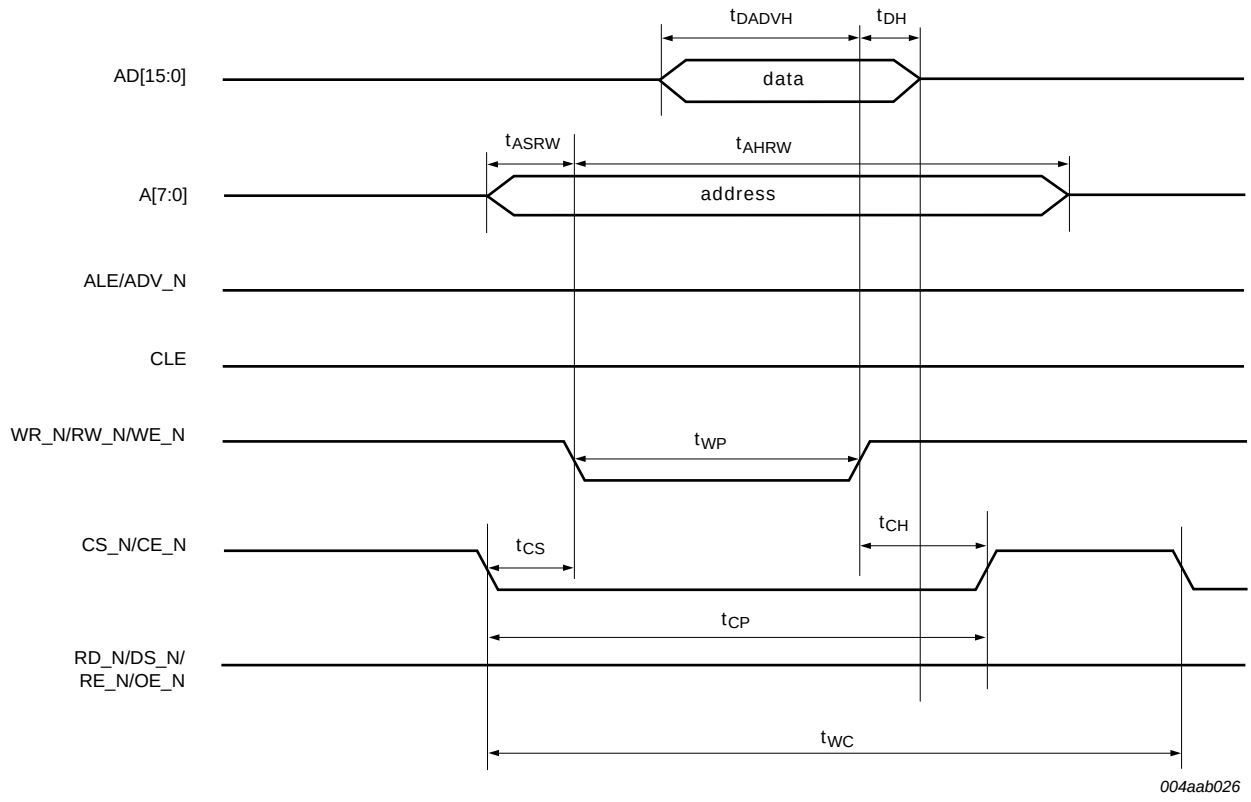



**Fig 12. Write in SRAM mode**

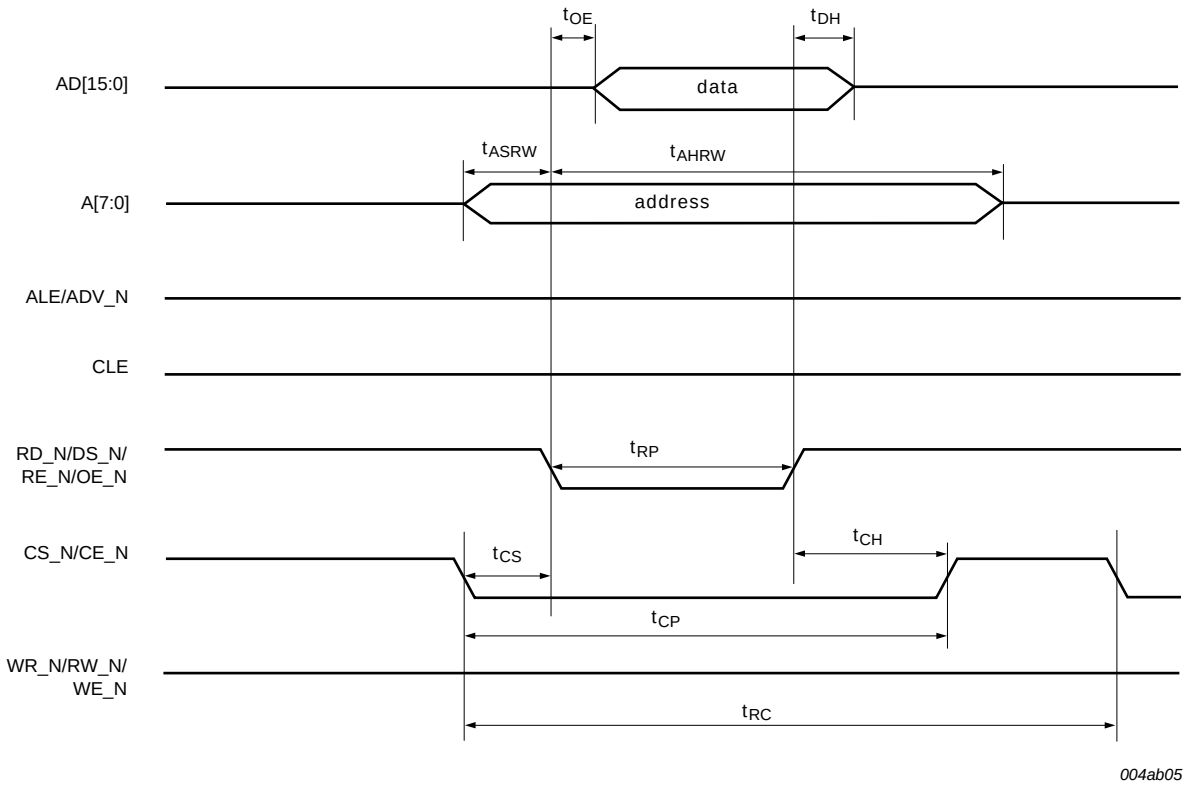



**Fig 13. Read in proprietary SRAM mode**

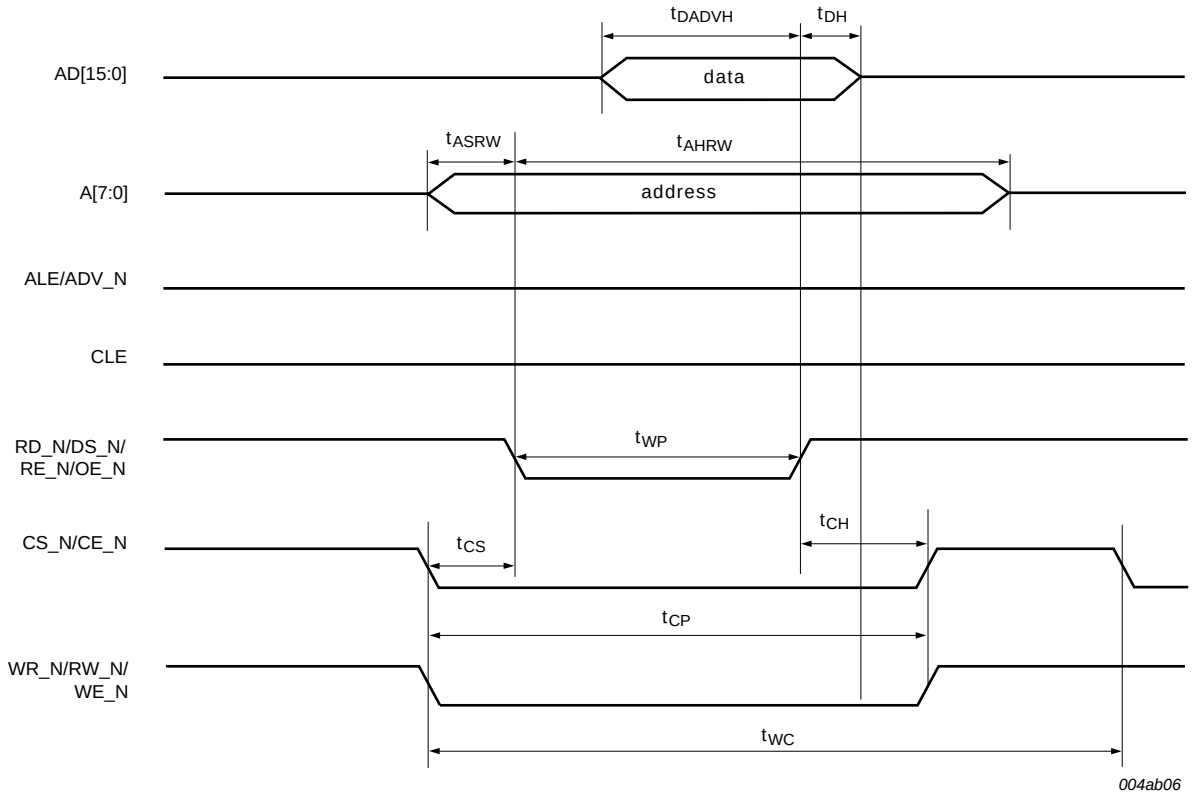



**Fig 14. Write in proprietary SRAM mode**

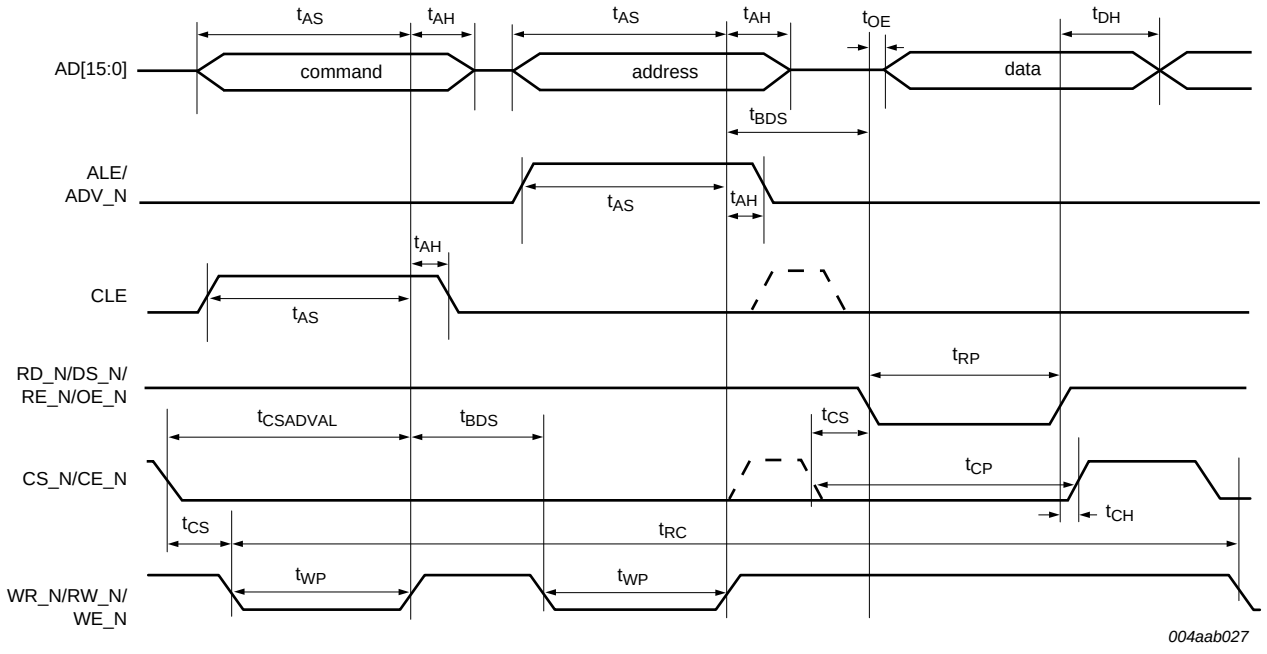



**Fig 15. Read in multiplex NAND mode**

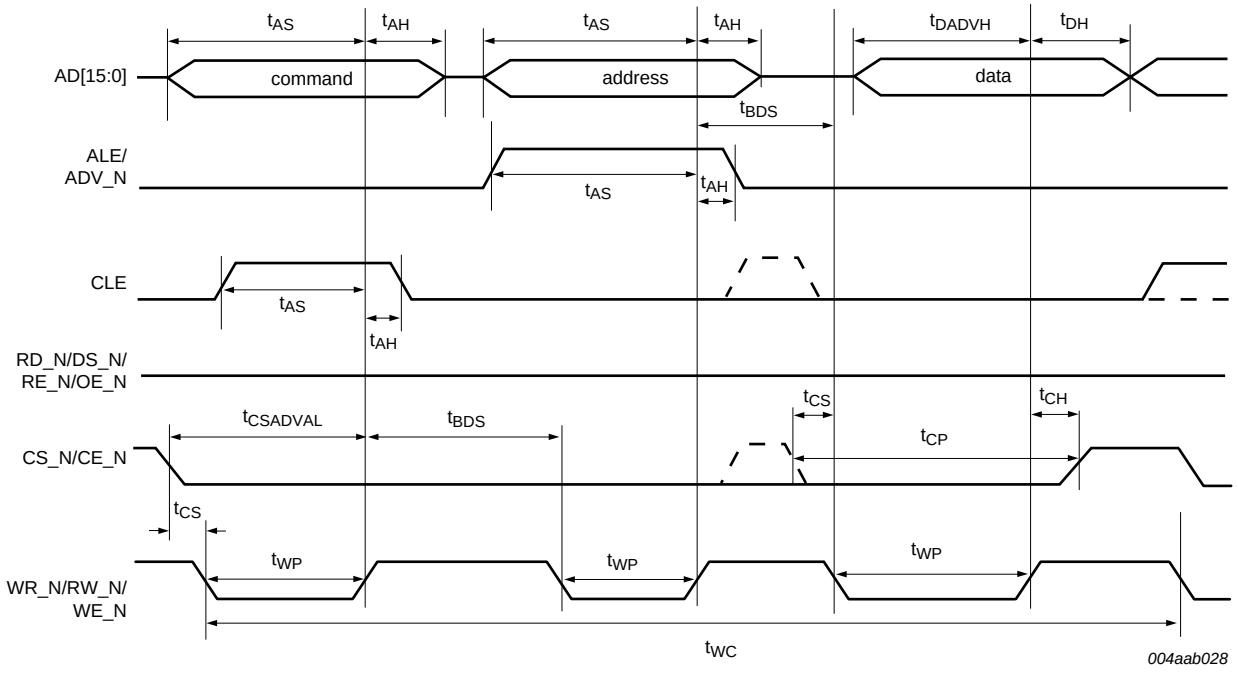



**Fig 16. Write in multiplex NAND mode**

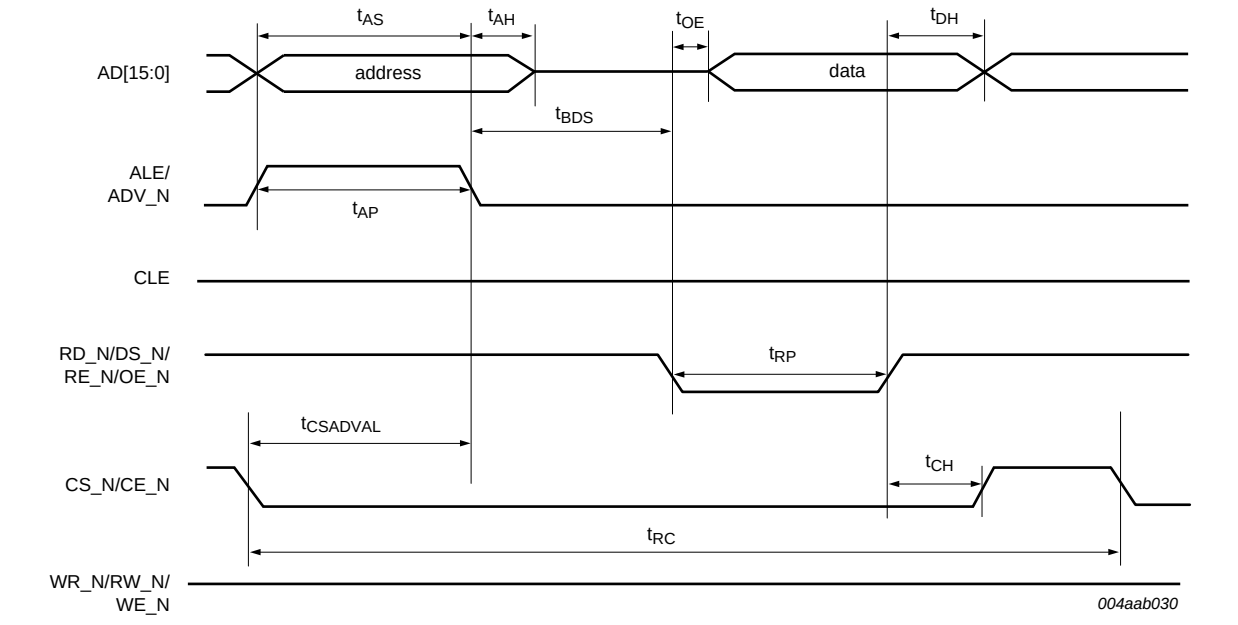



**Fig 17. Read in general multiplex mode**

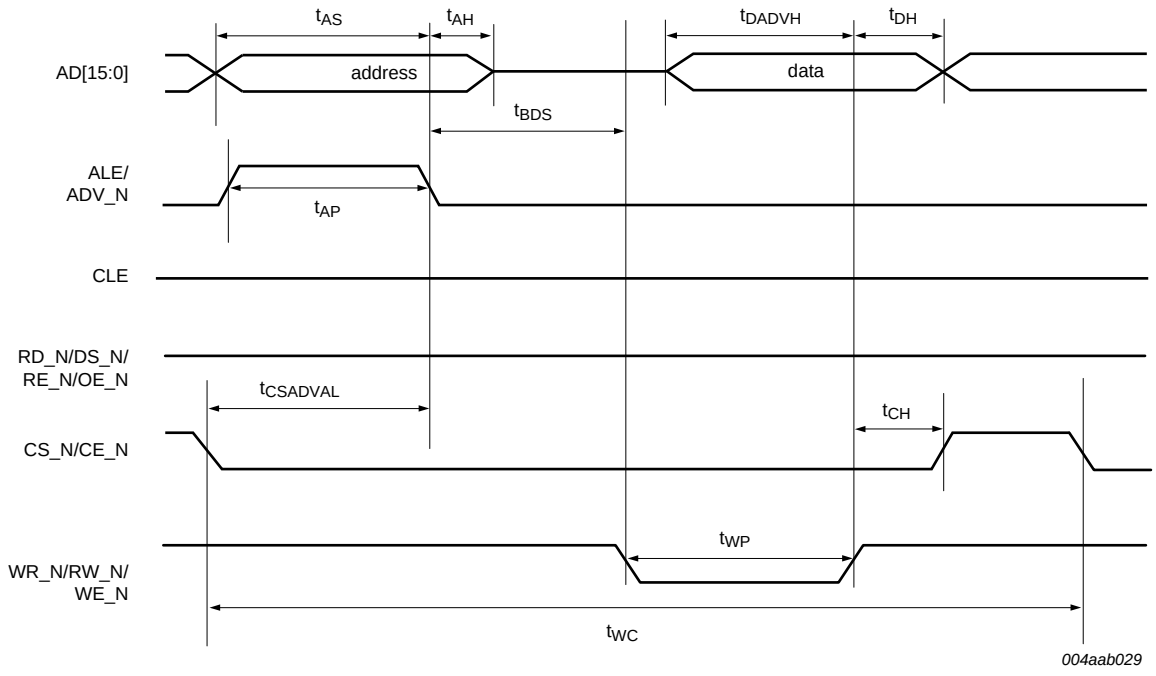



**Fig 18. Write in general multiplex mode**

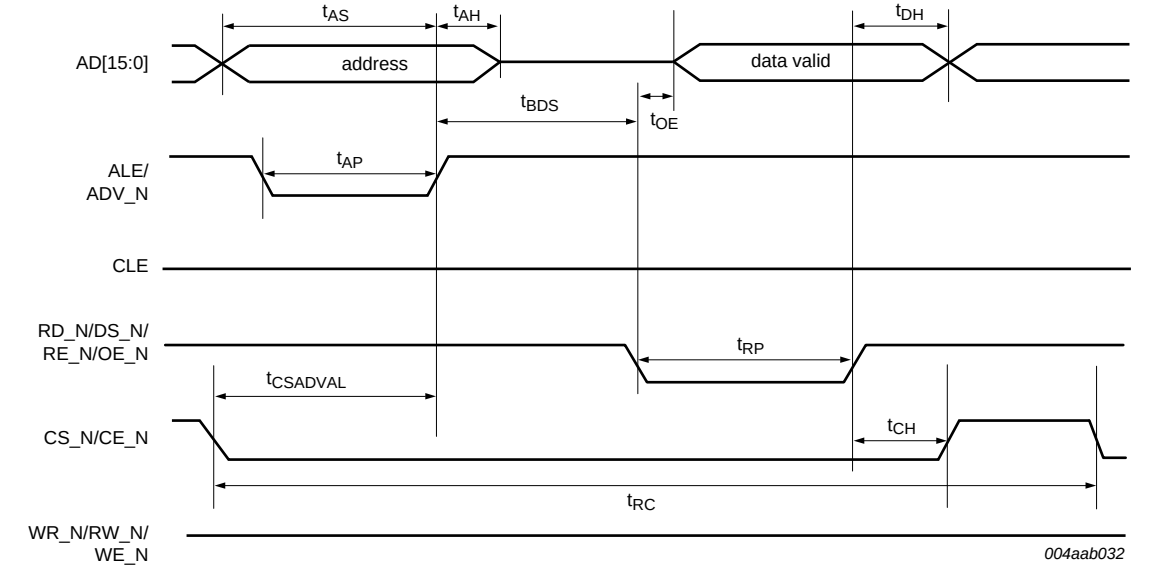



**Fig 19. Read in multiplex NOR mode**

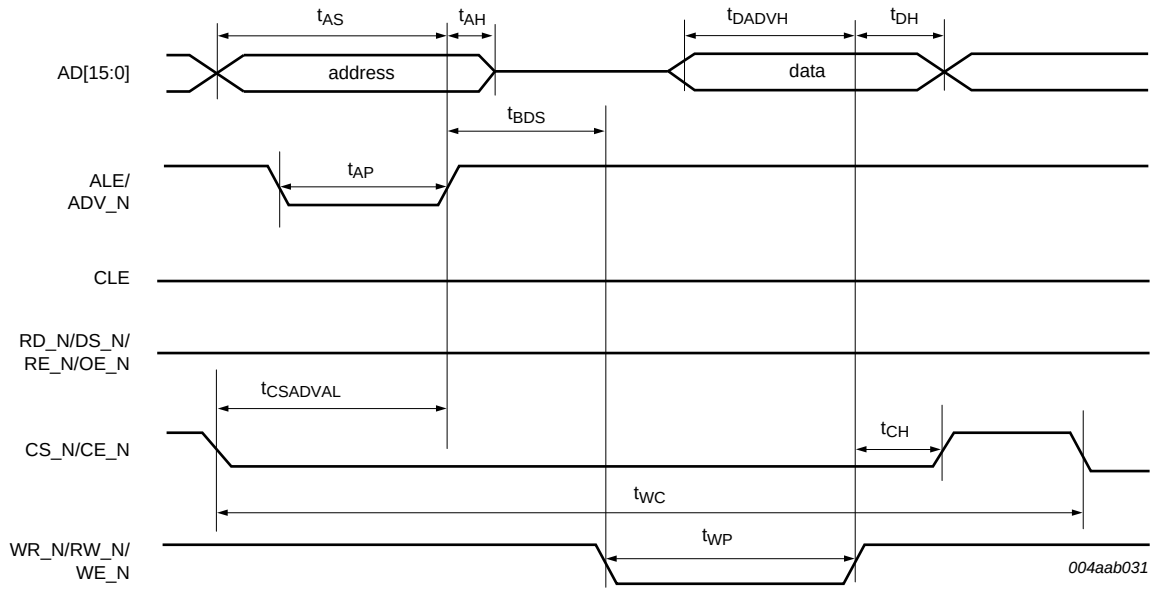



**Fig 20. Write in multiplex NOR mode**

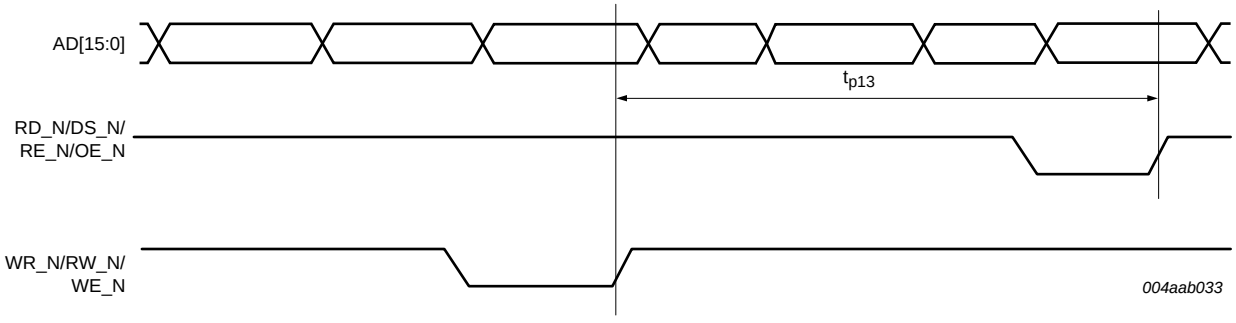
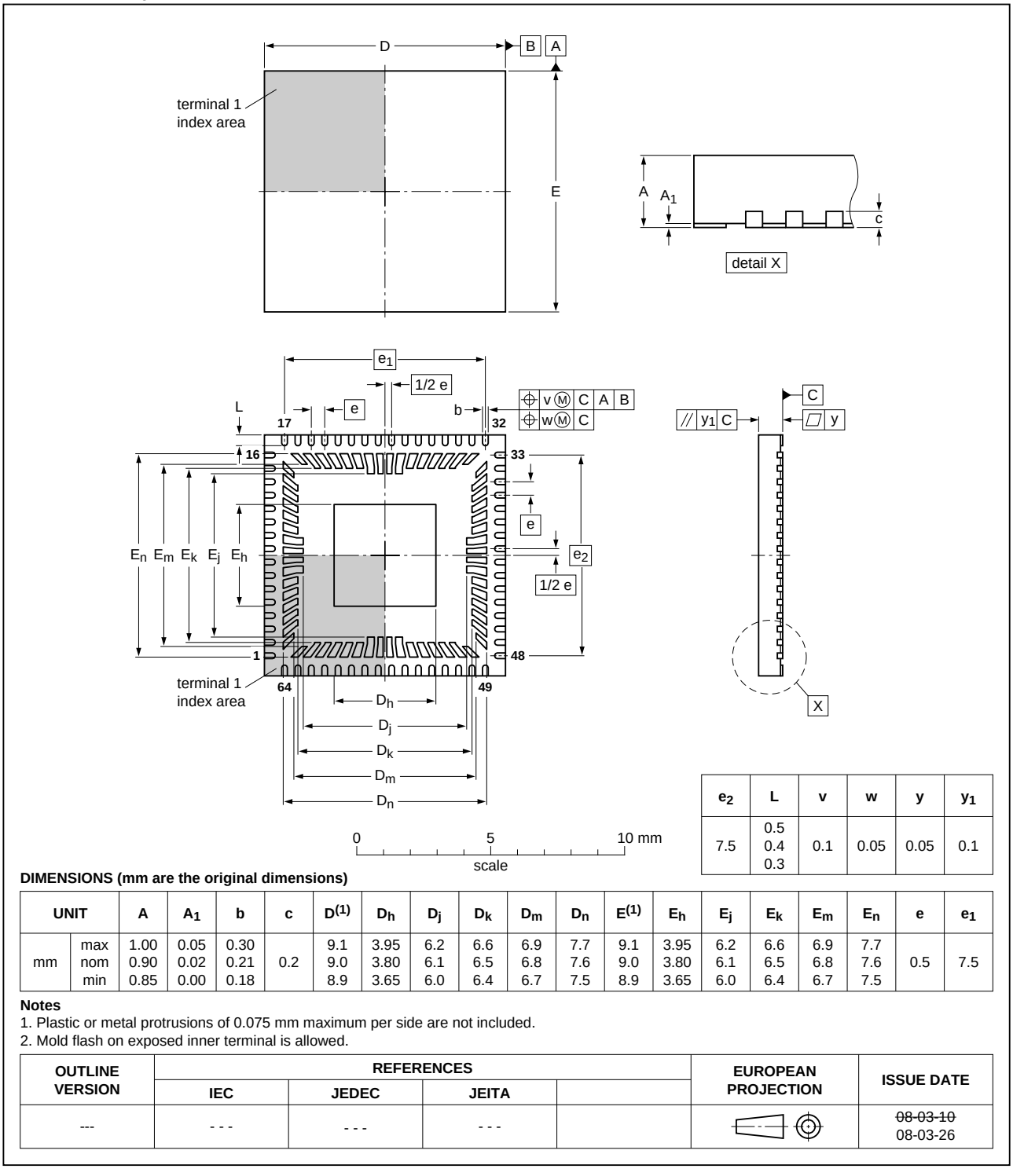



**Fig 21. Memory read prefetch**

# 16. Package outline

**VFQFPN64: plastic thermal enhanced very thin quad flat package; no leads; 64 terminals; body 9 x 9 x 1.0 mm**




| $e_2$ | L | v | w | y | $y_1$ |
|---|---|---|---|---|---|
| 7.5 | 0.5<br>0.4<br>0.3 | 0.1 | 0.05 | 0.05 | 0.1 |

**DIMENSIONS (mm are the original dimensions)**

| UNIT | | A | $A_1$ | b | c | $D^{(1)}$ | $D_h$ | $D_j$ | $D_k$ | $D_m$ | $D_n$ | $E^{(1)}$ | $E_h$ | $E_j$ | $E_k$ | $E_m$ | $E_n$ | e | $e_1$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| mm | max | 1.00 | 0.05 | 0.30 | | 9.1 | 3.95 | 6.2 | 6.6 | 6.9 | 7.7 | 9.1 | 3.95 | 6.2 | 6.6 | 6.9 | 7.7 | | |
| | nom | 0.90 | 0.02 | 0.21 | 0.2 | 9.0 | 3.80 | 6.1 | 6.5 | 6.8 | 7.6 | 9.0 | 3.80 | 6.1 | 6.5 | 6.8 | 7.6 | 0.5 | 7.5 |
| | min | 0.85 | 0.00 | 0.18 | | 8.9 | 3.65 | 6.0 | 6.4 | 6.7 | 7.5 | 8.9 | 3.65 | 6.0 | 6.4 | 6.7 | 7.5 | | |

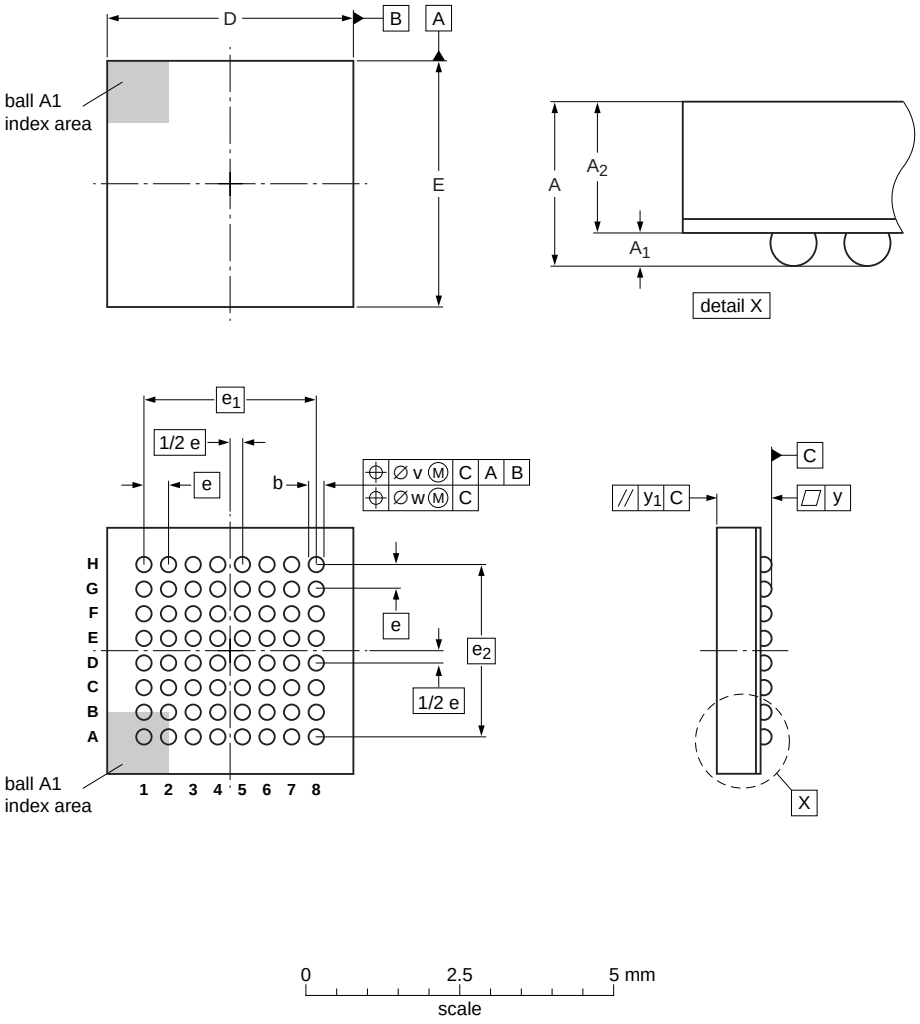
**Notes**

1. Plastic or metal protrusions of 0.075 mm maximum per side are not included.
2. Mold flash on exposed inner terminal is allowed.

| OUTLINE VERSION | REFERENCES | | | | EUROPEAN PROJECTION | ISSUE DATE |
|---|---|---|---|---|---|---|
| | IEC | JEDEC | JEITA | | | |
| --- | - - - | - - - | - - - | | | ~~08-03-10~~<br>08-03-26 |

**Fig 22. Package outline VFQFPN64**

**TFBGA64: plastic thin fine-pitch ball grid array package; 64 balls; body 4 x 4 x 0.8 mm**




**DIMENSIONS (mm are the original dimensions)**

| UNIT | A max | $A_1$ | $A_2$ | b | D | E | e | $e_1$ | $e_2$ | v | w | y | $y_1$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| mm | 1.1 | 0.25<br>0.15 | 0.85<br>0.75 | 0.3<br>0.2 | 4.1<br>3.9 | 4.1<br>3.9 | 0.4 | 2.8 | 2.8 | 0.15 | 0.05 | 0.08 | 0.1 |

| OUTLINE VERSION | REFERENCES | | | | EUROPEAN PROJECTION | ISSUE DATE |
|---|---|---|---|---|---|---|
| | IEC | JEDEC | JEITA | | | |
| --- | --- | --- | --- | | | ~~06-09-22~~<br>06-09-27 |

**Fig 23. Package outline TFBGA64**

# 17. Abbreviations

Table 162. Abbreviations

| Acronym | Description |
|---|---|
| ACK | Acknowledge |
| ATL | Asynchronous Transfer List |
| ATX | Analog USB Transceiver |
| DMA | Direct Memory Access |
| EHCI | Enhanced Host Controller Interface |
| FIFO | First In, First Out |
| HC | Host Controller |
| HNP | Host Negotiation Protocol |
| HS | High-Speed |
| INT | Interrupt |
| ISO | Isochronous |
| NAK | Not Acknowledge |
| OHCI | Open Host Controller Interface |
| OTG | On-The-Go |
| PCB | Printed-Circuit Board |
| PCI | Peripheral Component Interconnect |
| PID | Packet Identifier |
| PIO | Parallel I/O |
| PLL | Phase-Locked Loop |
| PTD | Proprietary Transfer Descriptor |
| SIE | Serial Interface Engine |
| SOF | Start-Of-Frame |
| SRP | Session Request Protocol |
| TT | Transaction Translator |
| USB | Universal Serial Bus |

# 18. References

[1] Universal Serial Bus Specification Rev. 2.0

[2] Enhanced Host Controller Interface Specification for Universal Serial Bus Rev. 1.0

[3] On-The-Go Supplement to the USB Specification Rev. 1.3

# 19. Revision history

Table 163. Revision history

| Revision | Release date | Data sheet status | Change notice |
|---|---|---|---|
| 2 | 20110224 | Product data sheet | - |
| Modifications: | • Table 153 “Dynamic characteristics: system clock”: updated conditions for $f_{clk}$. | | |
| 1 | 20100318 | Product data sheet | - |

# 20. Tables

Table 1. Ordering information . . . . . 3
Table 2. Marking codes . . . . . 3
Table 3. Pin description . . . . . 6
Table 4. Bus configuration modes . . . . . 11
Table 5. Pinning information of the bus interface . . . . . 12
Table 6. Register address . . . . . 15
Table 7. Memory address . . . . . 21
Table 8. Using the IRQ MASK AND or IRQ MASK OR registers . . . . . 24
Table 9. Endpoint access and programmability . . . . . 26
Table 10. Power supply . . . . . 28
Table 11. Overview of OTG controller-specific registers . 30
Table 12. OTG_CTRL - OTG Control register (address set: E4h, clear: E6h) bit allocation . . . . . 30
Table 13. OTG_CTRL - OTG Control register (address set: E4h, clear: E6h) bit description . . . . . 31
Table 14. OTG_INTR_SRC - OTG Interrupt Source register (address E8h) bit allocation . . . . . 32
Table 15. OTG_INTR_SRC - OTG Interrupt Source register (address E8h) bit description . . . . . 32
Table 16. OTG_INTR_L - OTG Interrupt Latch register (address set: ECh, clear: EEh) bit allocation . . 32
Table 17. OTG_INTR_L - OTG Interrupt Latch register (address set: ECh, clear: EEh) bit description . 33
Table 18. OTG_INTR_EN_F - OTG Interrupt Enable Fall register (address set: F0h, clear: F2h) bit allocation . . . . . 33
Table 19. OTG_INTR_EN_F - OTG Interrupt Enable Fall register (address set: F0h, clear: F2h) bit description . . . . . 33
Table 20. OTG_INTR_EN_R - OTG Interrupt Enable Rise register (address set: F4h, clear: F6h) bit allocation . . . . . 34
Table 21. OTG_INTR_EN_R - OTG Interrupt Enable Rise register (address set: F4h, clear: F6h) bit description . . . . . 34
Table 22. OTG_TMR - OTG Timer register (address low word set: F8h, low word clear: FAh; high word set: FCh, high word clear: FEh) bit allocation . . . . . 35
Table 23. OTG_TMR - OTG Timer register (address low word set: F8h, low word clear: FAh; high word set: FCh, high word clear: FEh) bit description . . . . 35
Table 24. Overview of host controller-specific registers . 36
Table 25. USBCMD - USB Command register (address 8Ch) bit allocation . . . . . 37
Table 26. USBCMD - USB Command register (address 8Ch) bit description . . . . . 37
Table 27. USBSTS - USB Status register (address 90h) bit allocation . . . . . 38
Table 28. USBSTS - USB Status register (address 90h) bit description . . . . . 38
Table 29. USBINTR - USB Interrupt register (address 94h) bit allocation . . . . . 38
Table 30. USBINTR - USB Interrupt register (address 94h) bit description . . . . . 39
Table 31. FRINDEX - Frame Index register (address: 98h) bit allocation . . . . . 39
Table 32. FRINDEX - Frame Index register (address: 98h) bit description . . . . . 40
Table 33. CONFIGFLAG - Configure Flag register (address 9Ch) bit allocation . . . . . 40
Table 34. CONFIGFLAG - Configure Flag register (address 9Ch) bit description . . . . . 41
Table 35. PORTSC1 - Port Status and Control 1 register (address A0h) bit allocation . . . . . 41
Table 36. PORTSC1 - Port Status and Control 1 register (address A0h) bit description . . . . . 41
Table 37. ISO PTD Done Map register (address A4h) bit description . . . . . 42
Table 38. ISO PTD Skip Map register (address A6h) bit description . . . . . 42
Table 39. ISO PTD Last PTD register (address A8h) bit description . . . . . 43
Table 40. INT PTD Done Map register (address AAh) bit description . . . . . 43
Table 41. INT PTD Skip Map register (address ACh) bit description . . . . . 43
Table 42. INT PTD Last PTD register (address AEh) bit description . . . . . 44
Table 43. ATL PTD Done Map register (address B0h) bit description . . . . . 44
Table 44. ATL PTD Skip Map register (address B2h) bit description . . . . . 44
Table 45. ATL PTD Last PTD register (address B4h) bit description . . . . . 45
Table 46. HW Mode Control - Hardware Mode Control register (address B6h) bit allocation . . . . . 45
Table 47. HW Mode Control - Hardware Mode Control register (address B6h) bit description . . . . . 45
Table 48. SW Reset - Software Reset register (address B8h) bit allocation . . . . . 46
Table 49. SW Reset - Software Reset register (address B8h) bit description . . . . . 47
Table 50. HcBufferStatus - Host Controller Buffer Status register (address BAh) bit allocation . . . . . 47
Table 51. HcBufferStatus - Host Controller Buffer Status

continued >>

register (address BAh) bit description . . . . . .48
Table 52. HcDMAConfiguration - Host Controller Direct Memory Access Configuration register (address BCh) bit allocation . . . . . .48
Table 53. HcDMAConfiguration - Host Controller Direct Memory Access Configuration register (address BCh) bit description . . . . . .49
Table 54. ATL Done Timeout register (address: C0h) bit description . . . . . .49
Table 55. Memory register (address C4h) bit allocation .50
Table 56. Memory register (address C4h) bit description 50
Table 57. Data Port register (address: C6h) bit description . . . . . .50
Table 58. Edge Interrupt Count register (address C8h) bit allocation . . . . . .50
Table 59. Edge Interrupt Count register (address C8h) bit description . . . . . .51
Table 60. DMA Data Port register (address 60h) bit description . . . . . .51
Table 61. DMA Start Address register (address CCh) bit description . . . . . .51
Table 62. Power Down Control register (address D0h) bit allocation . . . . . .51
Table 63. Power Down Control register (address D0h) bit description . . . . . .52
Table 64. HcInterrupt - Host Controller Interrupt register (address D4h) bit description . . . . . .53
Table 65. HcInterrupt - Host Controller Interrupt register (address D4h) bit description . . . . . .54
Table 66. HcInterruptEnable - Host Controller Interrupt Enable register (address D6h) bit allocation . .55
Table 67. HcInterruptEnable - Host Controller Interrupt Enable register (address D6h) bit description .56
Table 68. ISO IRQ MASK OR register (address D8h) bit description . . . . . .57
Table 69. INT IRQ MASK OR register (address DAh) bit description . . . . . .57
Table 70. ATL IRQ MASK OR register (address DCh) bit description . . . . . .57
Table 71. ISO IRQ MASK AND register (address DEh) bit description . . . . . .57
Table 72. INT IRQ MASK AND register (address E0h) bit description . . . . . .58
Table 73. ATL IRQ MASK AND register (address E2h) bit description . . . . . .58
Table 74. High-speed bulk IN and OUT: bit allocation . . .60
Table 75. High-speed bulk IN and OUT: bit description . .61
Table 76. High-speed isochronous IN and OUT: bit allocation . . . . . .64
Table 77. High-speed isochronous IN and OUT: bit description . . . . . .65
Table 78. High-speed interrupt IN and OUT: bit allocation . . . . . 68
Table 79. High-speed interrupt IN and OUT: bit description . . . . . 69
Table 80. Microframe description . . . . . 71
Table 81. Start and complete split for bulk: bit allocation 72
Table 82. Start and complete split for bulk: bit description . . . . . 73
Table 83. SE description . . . . . 75
Table 84. Start and complete split for isochronous: bit allocation . . . . . 76
Table 85. Start and complete split for isochronous: bit description . . . . . 77
Table 86. Start and complete split for interrupt: bit allocation . . . . . 80
Table 87. Start and complete split for interrupt: bit description . . . . . 81
Table 88. Microframe description . . . . . 83
Table 89. SE description . . . . . 83
Table 90. Peripheral controller-specific register overview 84
Table 91. ADDR - Address register (address 00h) bit allocation . . . . . 85
Table 92. ADDR - Address register (address 00h) bit description . . . . . 85
Table 93. MODE - Mode register (address 0Ch) bit allocation . . . . . 85
Table 94. MODE - Mode register (address 0Ch) bit description . . . . . 86
Table 95. INTR_CONF - Interrupt Configuration register (address 10h) bit allocation . . . . . 87
Table 96. INTR_CONF - Interrupt Configuration register (address 10h) bit description . . . . . 87
Table 97. Debug mode settings . . . . . 87
Table 98. DcInterruptEnable - Device Controller Interrupt Enable register (address 14h) bit allocation . . 88
Table 99. DcInterruptEnable - Device Controller Interrupt Enable register (address 14h) bit description . 88
Table 100. ENDP_INDEX - Endpoint Index register (address 2Ch) bit allocation . . . . . 90
Table 101. ENDP_INDEX - Endpoint Index register (address 2Ch) bit description . . . . . 90
Table 102. Addressing of endpoint buffers . . . . . 90
Table 103. CTRL_FUNC- Control Function register (address 28h) bit allocation . . . . . 90
Table 104. CTRL_FUNC- Control Function register (address 28h) bit description . . . . . 91
Table 105. DATA_PORT - Data Port register (address 20h) bit description . . . . . 92
Table 106. BUFFER_LENGTH - Buffer Length register (address 1Ch) bit description . . . . . 92
Table 107. DcBufferStatus - Device Controller Buffer Status

continued >>

register (address 1Eh) bit allocation . . . . . .93
Table 108. DcBufferStatus - Device Controller Buffer Status register (address 1Eh) bit description . . . . . .93
Table 109. ENDP_MAXPKTSIZE - Endpoint MaxPacketSize register (address 04h) bit allocation . . . . . .93
Table 110. ENDP_MAXPKTSIZE - Endpoint MaxPacketSize register (address 04h) bit description . . . . . .94
Table 111. ENDP_TYPE - Endpoint Type register (address 08h) bit allocation . . . . . .94
Table 112. ENDP_TYPE - Endpoint Type register (address 08h) bit description . . . . . .95
Table 113. Control bits for GDMA read/write (opcode = 00h/01h) . . . . . .96
Table 114. DMA_CMD - DMA Command register (address 30h) bit allocation . . . . . .96
Table 115. DMA_CMD - DMA Command register (address 30h) bit description . . . . . .96
Table 116. DMA commands . . . . . .96
Table 117. DMA_XFR_CTR - DMA Transfer Counter register (address 34h) bit allocation . . . . . .97
Table 118. DMA_XFR_CTR - DMA Transfer Counter register (address 34h) bit description . . . . . .98
Table 119. DcDMAConfiguration - Device Controller Direct Memory Access Configuration register (address 38h) bit allocation . . . . . .98
Table 120. DcDMAConfiguration - Device Controller Direct Memory Access Configuration register (address 38h) bit description . . . . . .98
Table 121. DMA_HW - DMA Hardware register (address 3Ch) bit allocation . . . . . .99
Table 122. DMA_HW - DMA Hardware register (address 3Ch) bit description . . . . . .99
Table 123. DMA_INTR_REASON - DMA Interrupt Reason register (address 50h) bit allocation . . . . . .99
Table 124. DMA_INTR_REASON - DMA Interrupt Reason register (address 50h) bit description . . . . . .100
Table 125. Internal EOT-functional relation with bit DMA_XFER_OK . . . . . .100
Table 126. Status of the bits in the DMA Interrupt Reason register . . . . . .100
Table 127. DMA_INTR_EN - DMA Interrupt Enable register (address 54h) bit allocation . . . . . .101
Table 128. DMA_INTR_EN - DMA Interrupt Enable register (address 54h) bit description . . . . . .101
Table 129. DMA_ENDP - DMA Endpoint register (address 58h) bit allocation . . . . . .101
Table 130. DMA_ENDP - DMA Endpoint register (address 58h) bit description . . . . . .102
Table 131. DMA_BRST_CTR - DMA Burst Counter register (address 64h) bit allocation . . . . . .102
Table 132. DMA_BRST_CTR - DMA Burst Counter register (address 64h) bit description . . . . . .102
Table 133. DcInterrupt - Device Controller Interrupt register (address 18h) bit allocation . . . . . .103
Table 134. DcInterrupt - Device Controller Interrupt register (address 18h) bit description . . . . . .103
Table 135. DcChipID - Device Controller Chip Identifier register (address 70h) bit description . . . . . .104
Table 136. FRAME_NO - Frame Number register (address 74h) bit allocation . . . . . 105
Table 137. FRAME_NO - Frame Number register (address 74h) bit description . . . . . 105
Table 138. SCRATCH - Scratch register (address 78h) bit allocation . . . . . 105
Table 139. SCRATCH - Scratch register (address 78h) bit description . . . . . 105
Table 140. UNLOCK_DEV - Unlock Device register (address 7Ch) bit allocation . . . . . 106
Table 141. UNLOCK_DEV - Unlock Device register (address 7Ch) bit description . . . . . 106
Table 142. INTR_PULSE_WIDTH - Interrupt Pulse Width register (address 80h) bit description . . . . . 106
Table 143. TEST_MODE - Test Mode register (address 84h) bit allocation . . . . . 106
Table 144. TEST_MODE - Test Mode register (address 84h) bit description . . . . . 107
Table 145. Limiting values . . . . . 108
Table 146. Recommended operating conditions . . . . . 108
Table 147. Static characteristics: supply pins . . . . . 109
Table 148. Static characteristics: digital pins . . . . . 109
Table 149. Static characteristics: USB interface block (pins DM1, DM2, DP1, DP2) . . . . . 109
Table 150. Static characteristics: $V_{BUS}$ comparators . . . 110
Table 151. Static characteristics: $V_{BUS}$ resistors . . . . . 111
Table 152. Static characteristics: ID detection circuit . . . 111
Table 153. Dynamic characteristics: system clock . . . . . 112
Table 154. Dynamic characteristics: power-up and reset 112
Table 155. Dynamic characteristics: digital pins . . . . . 112
Table 156. Dynamic characteristics: high-speed source electrical . . . . . 112
Table 157. Dynamic characteristics: full-speed source electrical . . . . . 113
Table 158. Dynamic characteristics: low-speed source electrical . . . . . 113
Table 159. DMA timing . . . . . 115
Table 160. PIO timing . . . . . 116
Table 161. PIO timing . . . . . 117
Table 162. Abbreviations . . . . . 127
Table 163. Revision history . . . . . 127

# 21. Figures

Fig 1. Block diagram . . . . . .4
Fig 2. Pin configuration TFBGA64 . . . . . .5
Fig 3. Pin configuration VFQFPN64 . . . . . .6
Fig 4. Write operation. . . . . .13
Fig 5. HNP sequence of events . . . . . .18
Fig 6. Internal hub . . . . . .20
Fig 7. Memory segmentation and access block diagram . . . . . .22
Fig 8. Power-up and reset sequence . . . . . .29
Fig 9. USB source differential data-to-EOP transition skew and EOP width . . . . . .114
Fig 10. DMA read and write . . . . . .115
Fig 11. Read in SRAM mode . . . . . .118
Fig 12. Write in SRAM mode . . . . . .119
Fig 13. Read in proprietary SRAM mode . . . . . .120
Fig 14. Write in proprietary SRAM mode . . . . . .121
Fig 15. Read in multiplex NAND mode . . . . . .121
Fig 16. Write in multiplex NAND mode . . . . . .122
Fig 17. Read in general multiplex mode . . . . . .122
Fig 18. Write in general multiplex mode. . . . . .123
Fig 19. Read in multiplex NOR mode . . . . . .123
Fig 20. Write in multiplex NOR mode. . . . . .124
Fig 21. Memory read prefetch . . . . . .124
Fig 22. Package outline VFQFPN64 . . . . . .125
Fig 23. Package outline TFBGA64 . . . . . .126

## 22. Contents


**1 General description . . . . . 1**
**2 Features . . . . . 1**
**3 Applications . . . . . 2**
3.1 Host or peripheral roles . . . . . 2
**4 Ordering information . . . . . 3**
**5 Marking . . . . . 3**
**6 Block diagram . . . . . 4**
**7 Pinning information . . . . . 5**
7.1 Pinning . . . . . 5
7.2 Pin description . . . . . 6
**8 Functional description . . . . . 11**
8.1 CPU bus interface . . . . . 11
8.2 Interface mode lock . . . . . 12
8.3 SRAM bus interface mode . . . . . 12
8.4 NAND bus interface mode . . . . . 13
8.5 NOR bus interface mode . . . . . 13
8.6 General multiplex bus interface mode . . . . . 14
8.7 DMA controller . . . . . 14
8.8 On-The-Go (OTG) controller . . . . . 16
8.8.1 Introduction . . . . . 16
8.8.2 Dual-role device . . . . . 16
8.8.3 Session Request Protocol (SRP) . . . . . 17
8.8.3.1 B-device initiating SRP . . . . . 17
8.8.3.2 A-device responding to SRP . . . . . 17
8.8.4 Host Negotiation Protocol (HNP) . . . . . 17
8.8.4.1 Sequence of HNP events . . . . . 18
8.8.5 Power saving in the idle state and during wake-up . . . . . 19
8.9 USB host controller . . . . . 19
8.9.1 ISP1763A USB host controller and hub internal architecture . . . . . 19
8.9.2 Host controller buffer memory block . . . . . 20
8.9.2.1 General considerations . . . . . 20
8.9.2.2 Structure of the host controller memory . . . . . 20
8.9.3 Interrupts . . . . . 22
8.10 USB peripheral controller . . . . . 24
8.10.1 Introduction . . . . . 24
8.10.2 Peripheral controller data transfer operation . 25
8.10.2.1 IN data transfer . . . . . 25
8.10.2.2 OUT data transfer . . . . . 25
8.10.3 Endpoint description . . . . . 25
8.10.4 Peripheral controller suspend . . . . . 26
8.10.5 Peripheral controller resume . . . . . 27
8.10.6 Remote wake-up . . . . . 27
8.11 Phase-Locked Loop (PLL) clock multiplier . . . 27
8.12 Power management . . . . . 27
8.12.1 Power supply . . . . . 27
8.12.2 Power modes . . . . . 28
8.12.2.1 Operation mode . . . . . 28
8.12.2.2 Suspend mode . . . . . 28
8.12.2.3 Deep sleep mode . . . . . 28
8.12.2.4 Power-down mode . . . . . 28
8.12.2.5 ISP1763A wake up . . . . . 28
8.12.2.6 Isolation mode . . . . . 29
8.12.3 Power-up and reset sequence . . . . . 29
8.12.4 ATX reference voltage . . . . . 29
**9 OTG controller-specific registers . . . . . 30**
9.1 OTG control register . . . . . 30
9.1.1 OTG Control register . . . . . 30
9.2 OTG interrupt registers . . . . . 31
9.2.1 OTG Interrupt Source register . . . . . 31
9.2.2 OTG Interrupt Latch register . . . . . 32
9.2.3 OTG Interrupt Enable Fall register . . . . . 33
9.2.4 OTG Interrupt Enable Rise register . . . . . 34
9.3 OTG Timer register . . . . . 35
9.3.1 OTG Timer register . . . . . 35
**10 Host controller-specific registers . . . . . 36**
10.1 EHCI operational registers . . . . . 37
10.1.1 USBCMD register . . . . . 37
10.1.2 USBSTS register . . . . . 37
10.1.3 USBINTR register . . . . . 38
10.1.4 FRINDEX register . . . . . 39
10.1.5 CONFIGFLAG register . . . . . 40
10.1.6 PORTSC1 register . . . . . 41
10.1.7 ISO PTD Done Map register . . . . . 42
10.1.8 ISO PTD Skip Map register . . . . . 42
10.1.9 ISO PTD Last PTD register . . . . . 43
10.1.10 INT PTD Done Map register . . . . . 43
10.1.11 INT PTD Skip Map register . . . . . 43
10.1.12 INT PTD Last PTD register . . . . . 43
10.1.13 ATL PTD Done Map register . . . . . 44
10.1.14 ATL PTD Skip Map register . . . . . 44
10.1.15 ATL PTD Last PTD register . . . . . 44
10.2 Configuration registers . . . . . 45
10.2.1 HW Mode Control register . . . . . 45
10.2.2 SW Reset register . . . . . 46
10.2.3 HcBufferStatus register . . . . . 47
10.2.4 HcDMAConfiguration register . . . . . 48
10.2.5 ATL Done Timeout register . . . . . 49
10.2.6 Memory register . . . . . 49
10.2.7 Data Port register . . . . . 50
10.2.8 Edge Interrupt Count register . . . . . 50
10.2.9 DMA Data Port register . . . . . 51


continued >>

10.2.10 DMA Start Address register . . . . . 51
10.2.11 Power Down Control register . . . . . 51
10.3 Interrupt registers . . . . . 53
10.3.1 HcInterrupt register . . . . . 53
10.3.2 HcInterruptEnable register . . . . . 55
10.3.3 ISO IRQ MASK OR register . . . . . 57
10.3.4 INT IRQ MASK OR register . . . . . 57
10.3.5 ATL IRQ MASK OR register . . . . . 57
10.3.6 ISO IRQ MASK AND register . . . . . 57
10.3.7 INT IRQ MASK AND register . . . . . 58
10.3.8 ATL IRQ MASK AND register . . . . . 58
10.4 Proprietary Transfer Descriptor (PTD) . . . . . 58
10.4.1 High-speed bulk IN and OUT . . . . . 60
10.4.2 High-speed isochronous IN and OUT . . . . . 64
10.4.3 High-speed interrupt IN and OUT . . . . . 68
10.4.4 Start and complete split for bulk . . . . . 72
10.4.5 Start and complete split for isochronous . . . . . 76
10.4.6 Start and complete split for interrupt . . . . . 80
**11 Peripheral controller-specific registers . . . . . 84**
11.1 Register access . . . . . 84
11.2 Initialization registers . . . . . 85
11.2.1 Address register . . . . . 85
11.2.2 Mode register . . . . . 85
11.2.3 Interrupt Configuration register . . . . . 86
11.2.4 DcInterruptEnable register . . . . . 87
11.3 Data flow registers . . . . . 89
11.3.1 Endpoint Index register . . . . . 89
11.3.2 Control Function register . . . . . 90
11.3.3 Data Port register . . . . . 91
11.3.4 Buffer Length register . . . . . 92
11.3.5 DcBufferStatus register . . . . . 93
11.3.6 Endpoint MaxPacketSize register . . . . . 93
11.3.7 Endpoint Type register . . . . . 94
11.4 DMA registers . . . . . 95
11.4.1 DMA Command register . . . . . 96
11.4.2 DMA Transfer Counter register . . . . . 97
11.4.3 DcDMAConfiguration register . . . . . 98
11.4.4 DMA Hardware register . . . . . 99
11.4.5 DMA Interrupt Reason register . . . . . 99
11.4.6 DMA Interrupt Enable register . . . . . 100
11.4.7 DMA Endpoint register . . . . . 101
11.4.8 DMA Burst Counter register . . . . . 102
11.5 General registers . . . . . 102
11.5.1 DcInterrupt register . . . . . 102
11.5.2 Chip ID register . . . . . 104
11.5.3 Frame Number register . . . . . 104
11.5.4 Scratch register . . . . . 105
11.5.5 Unlock Device register . . . . . 105
11.5.6 Interrupt Pulse Width register . . . . . 106
11.5.7 Test Mode register . . . . . 106
**12 Limiting values . . . . . 108**
**13 Recommended operating conditions . . . . . 108**
**14 Static characteristics . . . . . 109**
**15 Dynamic characteristics . . . . . 112**
15.1 Timing . . . . . 115
15.1.1 DMA . . . . . 115
15.1.2 PIO . . . . . 116
**16 Package outline . . . . . 125**
**17 Abbreviations . . . . . 127**
**18 References . . . . . 127**
**19 Revision history . . . . . 127**
**20 Tables . . . . . 128**
**21 Figures . . . . . 131**
**22 Contents . . . . . 132**

**Please Read Carefully:**

The contents of this document are subject to change without prior notice. ST-Ericsson makes no representation or warranty of any nature whatsoever (neither expressed nor implied) with respect to the matters addressed in this document, including but not limited to warranties of merchantability or fitness for a particular purpose, interpretability or interoperability or, against infringement of third party intellectual property rights, and in no event shall ST-Ericsson be liable to any party for any direct, indirect, incidental and or consequential damages and or loss whatsoever (including but not limited to monetary losses or loss of data), that might arise from the use of this document or the information in it.



Contact information at www.stericsson.com under Contacts

**www.stericsson.com**